

This Page Is Inserted by IFW Operations  
and is not a part of the Official Record

## **BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning documents *will not* correct images,  
please do not report the images to the  
Image Problem Mailbox.**

**THIS PAGE BLANK (USPTO)**



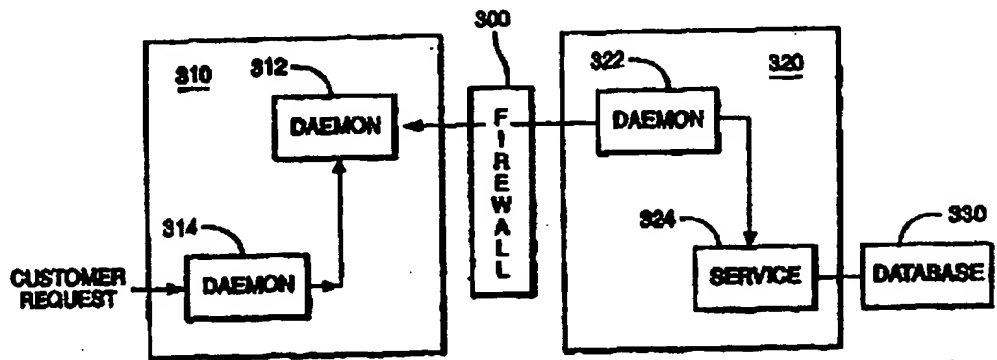
PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION  
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : <b>H04L 29/06</b>		<b>A1</b>	(11) International Publication Number: <b>WO 97/16911</b>
			(43) International Publication Date: 9 May 1997 (09.05.97)
(21) International Application Number: <b>PCT/GB96/00664</b>		(81) Designated States: BR, CA, CN, CZ, HU, JP, KR, PL, RU, European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).	
(22) International Filing Date: 20 March 1996 (20.03.96)			
(30) Priority Data: 08/551,260 ✓ 31 October 1995 (31.10.95) US		Published With international search report.	
(71) Applicant: INTERNATIONAL BUSINESS MACHINES CORPORATION [US/US]; Armonk, NY 10504 (US).			
(71) Applicant (for MC only): IBM UNITED KINGDOM LIMITED [GB/GB]; North Harbour, P.O. Box 41, Portsmouth, Hampshire PO6 3AU (GB).			
(72) Inventors: GORE, Robert, Cecil; 504 Cedar Lane, Pflugerville, TX 78660 (US). HAUGH, John, Frederick; 2302 Emmett Parkway, Austin, TX 78728 (US).			
(74) Agent: LING, Christopher, John; IBM United Kingdom Limited, Intellectual Property Dept., Hursley Park, Winchester, Hampshire SO21 2JN (GB).			

(54) Title: SECURED GATEWAY INTERFACE



(57) Abstract

A computer implemented method, uniquely programmed computer system, and article of manufacture embodying a computer readable program means allow a customer on an external network (310) to initiate an authorized business transaction utilizing internal business resources (330) on an internal network (320) without violating security firewalls (300). Specifically, the method directs an internal computer system (320) to allow an external computer system (310) to initiate a transaction request (2) using internal resources (330) without violating a security firewall (300) between the internal computer system (320) and the external computer system (310). The method includes a first step of authenticating a connection initiated by the internal computer system (320) between the internal computer system (320) and the external computer system (310), thereby establishing an authenticated connection. The second step includes calling by the external computer system (310) a transaction request (2) received by the external computer system (310). In response to calling the transaction request (2), the third step includes creating by the external computer system (310) a string comprising the transaction request (2), arguments, and process environment variables for executing the transaction request (2). The fourth step includes transmitting by the external computer system (310) the string to the internal computer system (320) through the authenticated connection. The fifth step includes verifying by the internal computer system (320) the transaction request (2). The sixth step includes recreating by the internal computer system (320) the original process environment. The final step includes executing by the internal computer system (320) the transaction request (2), thereby generating an output.

**THIS PAGE BLANK (USPTO)**

(19) 日本国特許庁 (J P)

(12) 公表特許公報 (A)

(11) 特許出願公表番号

特表平10-512696

(43) 公表日 平成10年(1998)12月2日

(51) Int.Cl. <sup>8</sup>	識別記号	F I	
G 0 6 F 13/00	3 5 1	G 0 6 F 13/00	3 5 1 Z
	3 5 5		3 5 5
15/00	3 1 0	15/00	3 1 0 A
H 0 4 L 9/32		G 0 9 C 1/00	6 6 0 E
12/28		H 0 4 L 11/20	B
審査請求 有 予備審査請求 有 (全 33 頁) 最終頁に続く			

(21) 出願番号 特願平9-517127  
 (86) (22) 出願日 平成8年(1996)3月20日  
 (85) 翻訳文提出日 平成10年(1998)4月20日  
 (86) 国際出願番号 PCT/GB96/00664 ✓  
 (87) 国際公開番号 WO97/16911 ✓  
 (87) 国際公開日 平成9年(1997)5月9日  
 (31) 優先権主張番号 08/551, 260 ✓  
 (32) 優先日 1995年10月31日  
 (33) 優先権主張国 米国 (US)  
 (81) 指定国 EP(AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), BR, CA, CN, CZ, HU, JP, KR, PL, RU

(71) 出願人 インターナショナル・ビジネス・マシーンズ・コーポレーション  
 アメリカ合衆国10504、ニューヨーク州アーモンク (番地なし)  
 (72) 発明者 ゴア、ロバート、セシル  
 アメリカ合衆国テキサス州ブルガーヴィル、セダー・レーン 504  
 (72) 発明者 ハウ、ジョン、フレデリック  
 アメリカ合衆国テキサス州オースチン、エメット・パークウェイ 2302  
 (74) 代理人 弁理士 坂口 博 (外1名)

(54) 【発明の名称】 保護されたゲートウェイ・インターフェース

## (57) 【要約】

コンピュータ読取り可能なプログラムを具体化する本発明のコンピュータ実施される方法、独自にプログラムされたコンピュータ・システム、及びその生産物は、外部ネットワークにおけるユーザが、セキュリティファイアウォールに違反することなく内部ネットワーク上の内部ビジネス資源を利用して、承認されたビジネス・トランザクションを開始することを可能にする。特定的に云えば、その方法は、外部コンピュータ・システム (310) が内部コンピュータ・システム (320) とその外部コンピュータ・システムとの間のセキュリティ・ファイアウォール (300) に違反することなく内部資源 (330) を使用してトランザクション・リクエスト (2) を開始させることを可能にするように、内部コンピュータ・システムに指示する。その方法は、内部コンピュータ・システムと外部コンピュータ・システムとの間の内部コンピュータ・システムによって開始される接続を認証し、それによって認証済み接続を確立するという第1ステップを含む。第2ステップは、外部コンピュータ・システムがその外部コンピュータ・システムによ

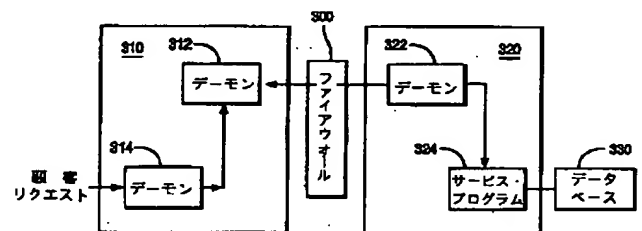


FIG. 3

**【特許請求の範囲】**

1. 内部コンピュータ・システム（320）と外部コンピュータ・システム（310）との間のセキュリティ・ファイアウォール（300）に違反することなく内部資源（330）を使用して前記外部コンピュータ・システムがトランザクション・リクエスト（2）を開始することを可能にするように前記内部コンピュータ・システムに指示するための方法にして、

前記内部コンピュータ・システムと前記外部コンピュータ・システムとの間の前記内部コンピュータ・システムによって開始される接続を認証し、それによって認証済み接続を確立するステップ（451）と、

前記外部コンピュータ・システムによって受け取られたトランザクション・リクエストを前記外部コンピュータ・システムによって呼び出すステップ（461）と、

前記トランザクション・リクエストの呼出しに応答して、プロセス環境変数を含むオリジナル・プロセス環境を前記外部コンピュータ・システムによって作成し、前記トランザクション・リクエストを実行するために前記トランザクション・リクエスト及び前記プロセス環境変数を含むストリングを作成するステップと

前記認証済み接続を通して前記内部コンピュータ・システムに前記ストリングを前記外部コンピュータ・システムによ

って送るステップ（435、436、437）と、

前記トランザクション・リクエストを前記内部コンピュータ・システムによって検証するステップ（454）と、

前記オリジナル・プロセス環境を前記内部コンピュータ・システムによって再作成するステップ（471）と、

前記トランザクション・リクエストを前記内部コンピュータ・システムによって実行し（472）、それによって出力を発生するステップ（482）と、

を含む方法。

2. （a）第1パスワード及び第1コミュニケーション・ポートを前記外部コンピュータ・システム（310）によって読み取るステップと、

(b) 前記第 1 コミュニケーション・ポートにおいて第 1 ソケットを前記外部コンピュータ・システムによって作成し、前記内部コンピュータ・システム (320) からの接続コールを前記第 1 ソケットにおいて傾聴するステップと、

(c) 第 2 パスワード及び第 2 コミュニケーション・ポートを前記内部コンピュータ・システムによって読み取るステップと、

(d) 前記第 2 コミュニケーション・ポートにおいて第 2 ソケットを前記内部コンピュータ・システムによって作成し、前記第 2 ソケットを通して前記外部コンピュータ・システムに接続コールを送り、それによって接続を確立するステップと、

を更に含む請求の範囲第 1 項に記載の方法。

3. 前記認証ステップは、

(e) 前記第 2 ソケットを通して前記外部コンピュータ・システム (310) に独自のタイムスタンプを前記内部コンピュータ・システム (320) によって送るステップと、

(f) 受け取られたタイム・スタンプでもって前記第 1 パスワードを前記外部コンピュータ・システムによってマングルするステップと、

(g) マングルされた第 1 パスワードを暗号化アルゴリズムでもって暗号化し、それによって第 1 パスワード・ストリングを作成するステップと、

(h) 前記第 1 パスワード・ストリングを前記内部コンピュータ・システムへ前記外部コンピュータ・システムによって送るステップと、

(i) 前記第 2 パスワードを使用してステップ (f) 乃至 (g) を前記内部コンピュータ・システムによって反復し、それによって第 2 パスワード・ストリングを作成するステップと、

(j) 前記第 1 パスワード・ストリングと前記第 2 パスワード・ストリングとを前記内部コンピュータ・システムによって比較するステップと、

を含む請求の範囲第 1 項に記載の方法。

4. 前記マングルするステップは、

前記タイムスタンプを 16 進数 0177 と論理的に AND

して独自の結果を生じさせるステップと、

前記固有の結果を前記第 1 パスワードの各文字と論理的に排他的 OR し、それによって前記マングルされた第 1 パスワードを生じさせるステップと、

を含む請求の範囲第 3 項に記載の方法。

5. 前記暗号化するステップは、

前記マングルされた第 2 パスワードの各文字をキーでもって暗号化し、それによって前記パスワード・ストリングを作成するステップ

を含む請求の範囲第 3 項に記載の方法。

6. 前記呼び出すステップは、

入力データ、アーギュメント、及びトランザクション・プログラムを実行するためのコマンドを含むトランザクション・リクエスト (2) を前記外部コンピュータ・システム (310) へ外部ネットワークによって送るステップと、

前記外部コンピュータ・システムが前記トランザクション・リクエストを受け取ることに応答して、前記プロセス環境変数を含むプロセス環境を第 1 デーモンによって定義するステップと、

を含む請求の範囲第 1 項に記載の方法。

7. 前記トランザクション・リクエスト (2) を前記外部コンピュータ・システム (310) によって呼び出すことに応答して、前記コマンドを呼び出すステップと、

前記コマンドを呼び出すことに応答して、スクリプトを実

行し、ユーザ入力データ、アーギュメント、及びトランザクション・リクエストを送るステップと、

前記スクリプトによって前記ストリングを作成するステップと、

を含み、前記ストリングは前記コマンド、アーギュメント、及び前記トランザクション・リクエストを実行するためのプロセス環境変数を更に含む請求の範囲第 6 項に記載の方法。

8. 前記ユーザ入力データ、前記外部コンピュータ・システムに常駐する第 2 デーモンに接続するための第 3 コミュニケーション・ポート、及びトランザクシ



ン・リクエスト（２）のタイプ及び前記ストリングを識別する識別子を前記外部コンピュータ・システム（３１０）に送って、前記外部コンピュータ・システムに常駐するクライアント・ルーチンを前記スクリプトによって呼び出すステップを更に含む請求の範囲第７項に記載の方法。

９．前記スクリプトによる呼び出しを受けることに応答して、前記クライアント・ルーチンを前記第２デーモンによって認証するステップと、

第３ソケット接続を子プロセスに送って第２デーモンによって分岐し、それによって、親プロセスが前記第１ソケット接続において前記内部コンピュータ・システムからの呼び出しを傾聴するようにするステップと、

前記クライアント・ルーチンを認証することに応答して、前記内部コンピュータ・システムに常駐する第３デーモンに

トランザクション・リクエストのタイプを前記子プロセスによって送るステップと、

を更に含む請求の範囲第８項に記載の方法。

１０．前記検証するステップは、

前記外部コンピュータ・システムにおけるメモリに記憶された有効サービステーブルを前記第３デーモンによって読み取るステップと、

前記子プロセスから受け取ったトランザクション・リクエストのタイプを前記有効サービス・テーブルと比較するステップと

を含み、前記タイプが前記有効サービス・テーブルにおいて見つかる場合、前記トランザクション・リクエストが検証される請求の範囲第１項に記載の方法。

１１．内部コンピュータ・システム（３２０）と外部コンピュータ・システム（３１０）との間のセキュリティ・ファイアウォール（３００）に違反することなく内部資源（３３０）を使用して前記外部コンピュータ・システムがトランザクション・リクエスト（２）を開始することを可能にするように前記内部コンピュータ・システムに指示するための独自にプログラムされたシステムにして、

前記内部コンピュータ・システムと前記外部コンピュータ・システムとの間の前記内部コンピュータ・システムによって開始された接続を認証し、それによっ

て認証済み接続を確立するための手段と、

前記外部コンピュータ・システムが受け取ったトランザクション・リクエストを前記外部コンピュータ・システムによって呼び出すための手段と、

前記トランザクション・リクエストの呼出しに応答して、プロセス環境変数を含むオリジナル・プロセス環境を前記外部コンピュータ・システムによって作成するための手段、及び前記トランザクション・リクエストを実行するために前記トランザクション・リクエスト、前記アーギュメント、及び前記プロセス環境変数を含むストリングを作成するするための手段と、

前記認証済み接続を通して前記内部コンピュータ・システムに前記ストリングを前記外部コンピュータ・システムによって送るための手段と、

前記トランザクション・リクエストを前記内部コンピュータ・システムによって検証するための手段と、

前記オリジナル・プロセス環境を前記内部コンピュータ・システムによって再作成するための手段と、

前記トランザクション・リクエストを前記内部コンピュータ・システムによって実行し、それによって出力を発生するための手段と、

を含むシステム。

12. コンピュータ読取り可能なプログラム・コード手段を具体化され、内部コンピュータ・システムと外部コンピュータ・システムとの間のセキュリティ・ファイアウォールに違

反することなく内部資源を使用して前記外部コンピュータ・システムがトランザクション・リクエストを開始することを前記内部コンピュータ・システムに可能にさせるためのコンピュータ使用可能媒体を含む生産品にして、前記生産品における前記コンピュータ読取り可能なプログラム・コード手段は、

前記内部コンピュータ・システムと前記外部コンピュータ・システムとの間の前記内部コンピュータ・システムによって開始された接続を認証し、それによって認証済み接続を確立するためのコンピュータ読取り可能プログラム手段と、

前記外部コンピュータ・システムが受け取ったトランザクション・リクエストを前記外部コンピュータ・システムによって呼び出すためのコンピュータ読取り可能プログラム手段と、

前記トランザクション・リクエストの呼出しに応答して、プロセス環境変数を含むオリジナル・プロセス環境を前記外部コンピュータ・システムによって作成するためのコンピュータ読取り可能プログラム手段、及び前記トランザクション・リクエストを実行するために前記トランザクション・リクエスト及び前記プロセス環境変数を含むストリングを作成するためのコンピュータ読取り可能プログラム手段と、

前記認証済み接続を通して前記内部コンピュータ・システムに前記ストリングを前記外部コンピュータ・システムによって送るためのコンピュータ読取り可能プログラム手段と、

前記トランザクション・リクエストを前記内部コンピュータ・システムによって検証するためのコンピュータ読取り可能プログラム手段と、

前記オリジナル・プロセス環境を前記内部コンピュータ・システムによって再作成するためのコンピュータ読取り可能プログラム手段と、

前記トランザクション・リクエストを前記内部コンピュータ・システムによって実行し、それによって出力を発生するためのコンピュータ読取り可能プログラム手段と、

を含むことを特徴とする生産品。

**【発明の詳細な説明】****保護されたゲートウェイ・インターフェース****技術分野**

本発明は、ネットワークのための保護されたゲートウェイ・インターフェースに関するものであり、更に詳しく、しかし、限定的でなく云えば、外部のネットワーク・ユーザが、セキュリティ・ファイアウォールに違反することなく内部資源を利用して、許可されたトランザクションを開始することを可能にするための保護されたゲートウェイ・インターフェースに関するものである。

**背景技術**

第1図は、従来技術の保護されたゲートウェイ・インターフェース（SGI）9を示す。そのSGIは、外部ネットワーク（例えば、インターネット）上の誰かからのユーザ／顧客リクエスト2を処理するための外部サーバ4（例えば、ハイパーテキスト転送プロトコル・デーモンHTTPD）、及び内部ネットワーク上の誰か（例えば、特定の企業のために働く誰か）及びデータベース8からのリクエストを処理するための内部サーバ7を有する。SGI 9は、更に、内部データベース8における内部トランザクションを外部から開始させることを防ぐためのファイアウォール6を含む。従って、

ファイアウォール6は、外部の顧客が内部ネットワーク（即ち、内部サーバ7及び内部データベース8）への直接接続を開始することを防ぐ。この制限は、顧客が外部ネットワークから内部トランザクションを簡単には開始させることができないので、製品及びサービスの購入リクエストのような有効なトランザクションも禁止してしまう。

上記の問題点に対する一般的な解決方法は、インバウンド・トラフィックに対する特定のポート（例えば、ポート84）をファイアウォール6において開くことを必要とする。しかし、この解決方法は、明らかに、内部ネットワークが外部の攻撃を免れ得ない状態にしておくことになる。別の解決方法は、すべての必要な資源（例えば、データベース8）を外部サーバ4上に設置することである。しかし、この解決方法は内部トランザクションの実行を禁止したままである。更に

、外部サーバ4はすべての必要な資源を保持するに十分な記憶装置を持たないないことがあり、或いは、それらの資源は機密性が高くて外部サーバ上に配置することができず、提供可能なサービスを制限することがある。

従って、セキュリティ・ファイアウォールに違反することなく内部ビジネス資源を利用して顧客が許可されたビジネス・トランザクションを開始することを可能にするための技法に対する大きな需要がある。

#### 発明の開示

従って、コンピュータ読取り可能なプログラム手段を具体化するコンピュータ実施方法、独自にプログラムされたコンピュータ・システム、及びその生産物は、外部ネットワークにおける顧客が、セキュリティ・ファイアウォールに違反することなく内部ネットワーク上の内部ビジネス資源を利用して、許可されたビジネス・トランザクションを開始することを可能にする。

特定の云えば、その方法は、外部コンピュータ・システムが、内部コンピュータ・システムとその外部コンピュータ・システムとの間のセキュリティ・ファイアウォールに違反することなく内部資源を使用してトランザクション・リクエストを開始することを可能にするように内部コンピュータ・システムに指示する。その方法は、内部コンピュータ・システムと外部コンピュータ・システムとの間の内部コンピュータ・システムによって開始される接続を認証し、それによって認証済み接続を確立する第1ステップを含む。第2ステップは、外部コンピュータ・システムがその外部コンピュータ・システムによって受信されたトランザクション・リクエストを呼び出すことを含む。第3ステップは、トランザクション・リクエストの呼出に応答して、外部コンピュータ・システムがそのトランザクション・リクエストとそのトランザクション・リクエストを実行するためのプロセス環境変数とを含むストリングを作成することを含む。第4ステップは、外部コンピュータ・システムが、そのストリングを、認証済み

接続を通して内部コンピュータ・システムに送ることを含む。第5ステップは、内部コンピュータ・システムがそのトランザクション・リクエストを検証するこ

とを含む。第6ステップは、内部コンピュータ・システムがオリジナル・プロセス環境を再作成することを含む。最後のステップは、内部コンピュータ・システムがそのトランザクション・リクエストを実行し、それによって出力を発生することを含む。

従って、本発明の目的は、ユーザ及び実際のトランザクション・プログラムにとって透明である保護されたゲートウェイ・インターフェースを作成することにある。

本発明のもう1つの目的は、ユーザが、内部資源を利用して、内部ネットワークに対するファイアウォールを通してトランザクションを有効に開始することを可能にすることにある。

本発明の更にもう1つの目的は、ユーザが、認証されたトランザクションの有効なセットのみを開始することを可能にすることにある。

本発明の更にもう1つの目的は、外部コンピュータ・システムがユーザからのトランザクション・リクエストを受け取る前に、内部コンピュータ・システムと外部コンピュータ・システムとの間の接続を安全に承認することにある。

本発明の更にもう1つの目的は、トランザクション・プログラムを修正する必要なくそれらをファイアウォール内に記憶することにある。

#### 図面の簡単な説明

添付図面を参照して、本発明を実施例によって説明することにする。

第1図は、本発明を実施する場合に使用するための通常のネットワーク・システムのブロック図を示す。

第2図は、本発明を実施するための代表的なハードウェア構成を示す。

第3図は、好適な実施例による保護されたゲートウェア・インターフェース（SGI）のブロック図を示す。

第4図は、第3図に示されたSGIの更に詳細なプロセス・フロー図を示す。  
発明を実施するための最良の形態

好適な実施例は、コンピュータ実施される方法、独自にプログラムされたコンピュータ・システム、及び、外部のユーザ／顧客が、セキュリティ・ファイアウ

オールに違反することなく内部ビジネス資源を利用して認証済みビジネス・トランザクションを開始することを可能にするように、内部コンピュータ・システムに指示するための詳細なロジックを具体化するメモリを含む。

本発明は、第2図に示されたコンピュータ・システムにおいて実施される。コンピュータ・システム100は、キャッシュ15、ランダム・アクセス・メモリ(RAM)14、読

取り専用メモリ(ROM)16、及び不揮発性RAM(NVRAM)32を処理するためのIBM社のPowerPC601又はIntel社の486マイクロプロセッサのような中央処理装置(CPU)10を含む。I/Oアダプタ18によって制御される1つ又は複数のディスク20が長期記憶装置を提供する。テープ、CD-ROM、及びWORMドライブを含む他の種々の記憶媒体が使用可能である。又、データ又はコンピュータ・プロセス命令を記憶するための取り外し可能な記憶媒体も具備可能である。(なお、IBM及びPowerPCはインターナショナル・ビジネス・マシーンズ・コーポレーションの商標であり、Intelはインテル・コーポレーションの商標である)。

Sun Solaris、マイクロソフト社のWindows NT、IBM社のOS/2、或いはアップル社のSystem7のような任意の適当なオペレーティング・システムのデスクトップからの命令及びデータがRAM14からCPU10を制御する。従って、そのデスクトップはRAM14から実行する。しかし、好適な実施例では、IBM RISC System/6000が、UNIXオペレーティング・システムのIBM社の実施形態であるAIXオペレーティング・システムを走らせる。しかし、前述のように、本発明を実施するために、他のハードウェア・プラットフォーム及びオペレーティング・システムを利用することが可能であることは当業者には容易に明らかであろう。(なお、Solaris

risはサン・マイクロシステムズ社の商標であり、System7はアップル・コンピュータ社の商標であり、OS/2、RISC System/6000

、及びAIXはIBM社の商標である。UNIXは、X/Open社を通して排他的に使用許諾された米国及びその他の国における商標である）。

ユーザは、ユーザ・インターフェース・アダプタ22によって制御されるI/O装置（即ち、ユーザ・コントロール装置）を通してコンピュータ・システム100とコミュニケーションする。ディスプレイ38はユーザに情報を表示し、一方、キーボード24、ポインティング装置26、及びスピーカ28はユーザがコンピュータ・システムに指示を行うことを可能にする。通信アダプタ34は、このコンピュータ・システムとネットワークに接続された他の処理装置との間の通信を制御する。ディスプレイ・アダプタ36は、このコンピュータ・システムとディスプレイ38との間のコミュニケーションを制御する。

第3図は、好適な実施例による保護されたゲートウェイ・インターフェース（SGI）のブロック図及びプロセス・フローを示す。SGIは一对のサーバ310及び320に常駐し、そのサーバの各々はコンピュータ・システム100（第1図参照）において実現される。外部サーバ310はファイアウォール300の外側に常駐し、一方、内部サーバ320はファイアウォール300の内側に常駐する。ファイアウォール300は、外部トランザクションがそれを通して内部サーバ320に達するのを防ぐ任意の適当な一般的なファイアウォールを使用して実現される。好適な実施例では、ファイアウォール300はネットワーク・ルータ（例えば、シスコ・ルータ）である。しかし、ファイアウォール300が内部サーバ320内に常駐してもよいことは当業者には容易に明らかであろう。

外部サーバ310は、インターネットのような外部のネットワークにおけるユーザ／顧客とのコミュニケーションを管理する。しかし、任意の公衆網又は専用網におけるSNA又はX.25のような任意のタイプの通信プロトコルが使用可能であることは当業者には明らかであろう。内部サーバ320は、内部コーポレート情報ネットワークのような内部ネットワーク上の内部資源（例えば、データベース330）のコミュニケーションを管理する。外部サーバ310は、アウトサイド・デーモン312を走らせ、一方、内部サーバ320はインサイド・デー



モン322を走らせ、それによって、ファイアウォール300にまたかるコミュニケーションを可能にする。デーモンは、外部事象を待ち、それらの事象が生じる時にいつも事前定義された一連のアクションを実行する長時間実行コンピュータ・プログラムである。デーモンはサービス・リクエストを傾聴し、リクエストされた時にそれらを遂行する。外部サーバ310は、外部ネットワークからのサービス・リクエストを傾聴するデーモン314も走らせる。内

部サーバ320は、所望の内部トランザクションを実行するためのサービス・プログラム324を含む。サービス・プログラム324及び内部データベース330は、ビジネス・トランザクション（以下で、更に詳細に説明される）を実施する一組のコンピュータ・プログラムを表す。

第4A図及び第4B図は、前に第3図に示したSGIの更に詳細なプロセス・フローを示す図である。外部サーバ310は、任意の適当な通常の通信プロトコル・デーモン（HTTPD）314、cgi-ビン415、SGI\_\_クライアント・ルーチン416、及びアウトサイド・デーモン（SGI\_\_OD）312を含む。アウトサイド・デーモン312は、SGI\_\_クライアント・ルーチン416及びインサイド・デーモン（SGI\_\_ID）322とコミュニケーションするためのクライアント／サーバ・ソフトウェアを含む。SGI\_\_クライアント・ルーチン416は、アウトサイド・デーモン312とコミュニケーションするためのクライアント／サーバ・ソフトウェアを含む。cgi-ビン415は、デーモン314によって実行されるソフトウェアのディレクトリである。特に、この例では、cgi-ビンは、SGI\_\_クライアント・ルーチン416とコミュニケーションするための特別パール・スクリプト（以下で、更に詳細に説明される）であるexample462を含む。好適な実施例では、デーモン314は、通常のハイパーテキスト転送プロトコル・デーモン（HTTPD）（一般には、ウェブ・サーバとしても知られてい

る）である。

内部サーバ320は、インサイド・デーモン322、サービス・プログラム3

24、及びc g iービン426を含む。サービス・プログラム324は、アウトサイド・デーモン312、インサイド・デーモン322とコミュニケーションし、c g iービン・ルーチン（例えば、example.pl 480）を実行する。この例では、example.pl 480は、ユーザ／顧客を承認するために及びビジネス・トランザクションを実行するために、内部コーポレート・データベース（例えば、第3図におけるコーポレート・データベース330）とコミュニケーションする。

顧客／ユーザが410においてトランザクションをうまくリクエストし得る前に、内部サーバ320及び外部サーバ310は適正に接続しなければならない。それを行うために、外部オペレーティング・システムはアウトサイド・デーモン312を実行して、外部サーバ310におけるファイルシステム（図示されてない）に常駐するパスワード・ファイルのコミュニケーション・ポート及びロケーションを識別する。一方、アウトサイド・デーモン312はパスワード・ファイルから8文字パスワードを読み取り、その識別されたポートにおいてソケットを作成し、そのソケットにおいてインサイド・デーモン322からの接続コールを傾聴する。従って、アウトサイド・デーモン312はサーバの役割を負い、430において、クライアントの役割を負ったインサイド・デー

モン322からの接続コールを待つ。更に、アウトサイド・デーモン312は第2ポートにおいてソケットを作成し（デーモン312のコミュニケーション・ポート+1）、432においてSGI\_クライアント・ルーチン416からの接続試行（以下で、更に詳細に説明される）を待つ。

内部オペレーティング・システムはインサイド・デーモン322を実行して、インサイド・デーモン322をアウトサイド・デーモン312に接続するためのコミュニケーション・ポート、外部サーバ310のホスト名、内部サーバ320におけるファイル・システム（図示されてない）に常駐するパスワード・ファイルのロケーション、及び内部サーバ320におけるファイル・システム（図示されてない）に常駐する有効サービス・ファイルのロケーションを識別する。一方、インサイド・デーモン322はパスワード・ファイルから8文字パスワードを

読み取り、サービス・ファイルを読み取って有効サービスのテーブルをメモリに記憶し、識別されたコミュニケーション・ポートにおいてソケットを作成し、最後に、450において、430で傾聴しているアウトサイド・デーモン312にファイアウォール300を越えて標準的な接続コールを発生する。その接続は内部サーバから開始されようとするので、ファイアウォール300はその接続を許容する。

インサイド・デーモン322及びアウトサイド・デーモン312が接続した後、インサイド・デーモン322及びアウ

トサイド・デーモン312は相互に適正に認証しなければならない。それを行うために、インサイド・デーモン322は、現在のタイムスタンプを検索するために内部オペレーティング・システムへのコールを開始し、そのタイムスタンプをアウトサイド・デーモン312に送り、それに応答して認証ストリングを待つ。アウトサイド・デーモン312はそのタイムスタンプを受け取り、そしてインサイド・デーモン322によって与えられたタイムスタンプでもってその8文字パスワードをマングルすること（後述の変更すること）によって認証ストリングを作成し、このマングルされた文字ストリングを標準的なUNIXクリプト・コマンド（又は、DESのような任意の適当な暗号化アルゴリズム）でもって暗号化し、そして、431において、その結果生じた認証ストリングをインサイド・デーモン322に送る。下記のC言語は、8文字パスワードをタイムスタンプでもってマングルするプロセスを示す。この“create\_auth”コードは3つのアーギュメントを必要とする。それらのアーギュメントのその第1はタイムスタンプ（即ち、auth\_time）であり、その第2はパスワード（即ち、“cred”）であり、その第3は発生された認証ストリングを記憶するためのバッファである。

```
int create_auth (time_t, char * cred * p)
{
    char  buf[9]  /* 一時的バッファ */
```

```
int    i;

bzero (buf, sizeof(buf)); /* バッファをクリアす
                           る */
strcpy (buf, cred); /* パスワードをバッファにロ
                     ードする */

/* バッファの各文字をマングルする */

for (i = 0; i < 8; i++) {
    buf[i] ^= (auth_time & 0177); /* タイムスタ
                                   ンプを論理的にANDし、結果を各文
                                   字と排他的にORする；各反復時にタ
                                   イムスタンプを修正する */
    auth_time >>= 4; /* タイムスタンプをビット単
                     位移動する */
{
for (i = 0; i < 8; i++)
    if (buf[i] == 0) /* 有効文字ストリングはヌル
                     を含むことができないので、
                     すべてのヌルを1に変更す
                     る */

        buf[i] = 1;
```

```
strcrp (p, cryt (buf, "aa") + 2); /* キーの代わ  
りに s s を使用してバッ  
ファを暗号化する */  
/* 暗号化結果のうちの (キー  
a a である) 最初の 2 文字  
をスキップする */  
/* 暗号化結果を、P によって  
指定されたユーザ供給バッ  
ファにコピーする */  
  
return 0;  
}
```

インサイド・デーモン322が同様にそのパスワードをタイムスタンプでもってマングルし、それを暗号化し、そしてそれをアウトサイド・デーモン312によって与えられた認証ストリングと比較する。それらの認証ストリングが一致する時、プロセスは反転され、アウトサイド・デーモン312が同様にインサイド・デーモン322を認証する（即ち、外部のオペレーティング・システムから新しいタイムスタンプを得て、そのタイムスタンプをインサイド・デーモン322に送り、インサイド・デーモン322はそのパスワードをその新しいタイムスタンプでもってマングルし、それを暗号化し、そして有効化するためにそれをアウトサイド・デ

ーモン312に戻す）。

この認証プロセスは、アウトサイド・デーモン312及びインサイド・デーモン322が共に知っている8文字パスワード、タイムスタンプによってランダム化された文字マングリング機能、及び暗号化プロセスを使用する。マングリング機能のために、上記のプロセスは、各認証及びすべてのトランザクションに対して異なる暗号化された認証ストリングを生じる。これは、捕捉された認証ストリ

ングがその後の如何なるトランザクションに対しても価値がないので、攻撃に対するその脆弱性をかなり減じる。

インサイド・デーモン322及びアウトサイド・デーモン312が相互に認証した後、前にクライアントとして作用したインサイド・デーモン322は今やサーバの役割を負い、452において、アウトサイド・デーモン312が453でサービス・ストリングを供給するのを待つ。アウトサイド・デーモン312は第2の指定されたポートにおいてもう1つのソケットを作成し、432においてSGI\_\_クライアント・ルーチン416からの接続試行を待つ（傾聴する）。従って、アウトサイド・デーモン312は、インサイド・デーモン322に関する擬似クライアント（情報はそれらの間で送られる）及びSGI\_\_クライアント・ルーチン416に関するサーバという2つの役割を負う。

デーモン314は、今や、顧客リクエスト410を受け付けるように準備される。顧客リクエストは、例えば、特定の

株式又は金融市場に関する調査情報を購入するためのトランザクションであってもよい。410において、顧客は、httpクライアント・アプリケーション・ユーザ・インターフェースを走っている顧客のシステムにおける特定のアイコン又は強調表示されたフレーズ上でクリックすることにより、次のようなトランザクション・リクエストを実行することを決定する。

```
http://external_server/cgi-bin/example.pl?stock1+stock2
```

そのhttpクライアント・ユーザ・インターフェースは、一般に、詳細なトランザクション情報（例えば、株式又は金融市場情報）及び課金情報（例えば、クレジット・カード番号）をユーザに尋ねる。ユーザは、リクエストされたサービスが単に認証されたユーザに与えられるだけである場合、そのユーザのユーザid及びパスワードを入力するように要求されることもある。

送られたユーザ入力のフォーマットは、トランザクションを実施するために使用されたハイパ・テキスト・マークアップ言語（HTML）フォームのタイプに依存する。2つのタイプの一般的なHTMLフォームが存在する。「GET」タ

イプはコマンド・ライン上にすべてのユーザ入力を配置する。従って、株式1 (stock1)、株式2 (stock2)、及び他の任意のユーザ入力下記のようにコマンド・ラインの一部となるであろう。

```
.../cgi-bin/example.pl?stock1+stock2+chargecardnumbe  
r+expdate
```

しかし、コマンド・ラインはネットワークを通してクリア・テキストとして送られるので、顧客のクレジット・カード番号及び有効期限日をネットワークを通して送ることは適切ではない。従って、暗号化を伴うHTMLの「PUT」タイプは、クレジット・カード番号及び有効期限日がネットワークを通して安全に送られるように使用される。この情報をすべて供給した後、httpクライアント・アプリケーションは、410において、httpを介して外部サーバ310にリクエストを送る。

460では、デーモン314が、一般に知られそして導入されているHTTP認証技法（例えば、顧客のパスワードを標準的なUNIXクリプト・コマンドをもって暗号化し、その結果を、デーモン314に常駐するhttpパスワード・ファイルにおけるパスワード・エントリと比較する）に従って顧客のパスワードを認証する。ユーザid及びパスワードが有効である場合、461において、デーモン314は「PUT」フォームを認識し、文字ストリームを自動的に非暗号化し、適正なUNIXプロセス環境を作成する。デーモン314は、PATH、USERNAME、LOGNAME、及びAUTHTYPE変数を含む標準的なUNIXプロセス環境を作成するための一般に知られた通常のhttp構成ファイル（図示されてない）を含む。その後、httpsvcl

70は（後述の）471においてこのプロセス環境を再作成する。一旦、プロセス環境が作成されてしまうと、デーモン314は、example.pl 462（cgiービン415に常駐しなければならない）を実行して任意の必要なアークギュメント（例えば、stock1及びstock2）及び例1.pl 462

の標準的な入力ストリームへのユーザ入力を送る。

`example.pl 462`が`cgi-bin 415`に常駐すると仮定すると、`example.pl 462`は内部データベース330（第3図参照）と直接にコミュニケーションし、所望のトランザクションを遂行するであろう。しかし、ファイアウォール300が存在し、`example.pl 462`が内部データベース330と直接にコミュニケーションしないようにするので、`example.pl 462`は実際のトランザクション・プログラムではない。むしろ、実際にトランザクション・プログラムは、ファイアウォール300の内側にある`example.pl 480`として`cgi-bin 426`に常駐する。従って、`cgi-bin 415`は、`cgi-bin 426`に常駐する実際のトランザクション・プログラムを実行する同じコマンドを使用して実行される「特別」パール・スクリプト（例えば、`example.pl 462`）を含む。別の方法として、各サービスが同じ方法で`SGI__クライアント・ルーチン 416`を呼び出すために「特別」パール・スクリプトを必要とする多くのサービスを、外部サーバ310

が提供する時、`example.pl 462`は、`cgi-bin 415`に常駐する単一のパール・スクリプトに対する象徴的なリンク（即ち、間接的ファイル名参照）となり得る。更に重大なことには、顧客にとって利用可能なリクエストは、それぞれ`cgi-bin 415`及び`cgi-bin 426`に常駐するパール・スクリプト及び対応するトランザクション・プログラムに限定される。

パール・スクリプト`example.pl 462`は、それに送られたデーモン314からプロセス環境へのすべてのアーギュメントを配置し（例えば、`SGI ARG1=stock1`；`SGI ARG2=stock2`）、その名前（その名前によってそれが呼び出される。この場合、`example.pl`）をプロセス環境に配置し（例えば、`SGI CMD=example.pl`）、`UNIX env`コマンド（プロセス環境変数をダンプする）を実行し、最後に、すべてのプロセス環境変数をヘッダ・ストリングに配置する。今や、ヘッダ・ストリングは、例えば、以下のように見える。



```
"PATH=/bin:/usr/bin\nAUTHTYPE=PEM\nUSERNAME=JohnDoe\nS
GIARG1=stock1\nSGIARG2=stock2\n=SGICMD=example.pl").
```

次に、463において、パール・スクリプト `example.pl 462` は、外部オペレーティング・システムを呼び出してその指定された第2ポート（デーモン312のコミュ

ニケーション・ポート+1）を検索させ、SGI\_\_クライアント・ルーチン416を実行し、リクエストされたサービスのタイプ（例えば、`httpsvc`）、指定された第2ポート、外部サーバのホスト名、ヘッダ・ストリング、及び顧客のユーザidを送る。又、`example.pl 462` は、任意の標準的な入力文字ストリーム（例えば、ユーザ入力のテキスト）をSGI\_\_クライアント・ルーチン416に標準的な入力として送る。その後、`example.pl 462` は、469において、SGI\_\_クライアント・ルーチン416から受け取ったすべての出力をデーモン314に送るであろう。

SGI\_\_クライアント・ルーチン416が463においてそれに送られた情報を使用して実行する時、SGI\_\_クライアント・ルーチン416はアウトサイド・デーモン312への認証された接続を確立する。それを行うために、417において、SGI\_\_クライアント・ルーチン416は、外部サーバ310に常駐する専用のクライアント・パスワード・ファイル（図示されてない）から8文字パスワードを読み取り、432において第2ソケット接続から傾聴しているアウトサイド・デーモン312への接続を、指定された第2ポートにおいて確立する。433において、アウトサイド・デーモン312はそれ自身のコピーを作成し、それを実行する（例えば、UNIXプロセス・フォーク）。親プロセスは子プロセスにソケット接続を与え、430に戻ってインサイド・デー

モン322からの別のコールを待つ。

434において、子プロセスはSGI\_\_クライアント・ルーチン416を認証する。それを行うために、アウトサイド・デーモン312もまた、外部サーバ3

10に常駐する専用クライアント・パスワード・ファイル（図示されてない）から8文字パスワードを読み取る。アウトサイド・デーモン312は、現在のタイムスタンプを検索するために外部オペレーティング・システムへのコールを開始し、432においてそのタイムスタンプをSGI\_\_クライアント・ルーチン416に送り、それに応答して認証ストリングを待つ。SGI\_\_クライアント・ルーチン416はそのタイムスタンプを受け取り、アウトサイド・デーモン312によって供給されたタイムスタンプでもってその8文字パスワードをマングルすることによって認証ストリングを作成し、このマングルされた文字ストリングを標準的なUNIX暗号コマンドでもって暗号化し、しかる後、434において、その結果生じた認証ストリングをアウトサイド・デーモン312に送る。アウトサイド・デーモン312は、同様にそのパスワードをタイムスタンプでもってマングルし、それを暗号化し、それをSGI\_\_クライアント・ルーチン416によって供給された認証ストリングと比較する。それらの認証ストリングが一致する場合、SGI\_\_クライアント・ルーチン416は認証される。

419において、認証が成功する場合、SGI\_\_クライア

ント・ルーチン416は、リクエストされたサービスのタイプをアウトサイド・デーモン312に送る。この例では、SGI\_\_クライアント・ルーチン416は、そのルーチン416がHPPTデーモン314によって間接的に呼び出されるので、いつもHTTPサービスをリクエストする。特別のパール・スクリプト（即ち、example.pl 462）は、リクエストされたサービスが「httpsvc」であることを表すアーギュメントを使用してSGI\_\_クライアント・ルーチン416を前に実行した。一方、アウトサイド・デーモン322は、435において、インサイド・デーモン322に「httpsvc」サービス・リクエストを送る。

452において、インサイド・デーモン322は、アウトサイド・デーモン312からのサービス・リクエストが受け取られるのを待つ。453において、インサイド・デーモン322は、アウトサイド・デーモン312からのサービス・リクエストを受け取り、それ自身の複写イメージを作成し、それを実行する（例

例えば、UNIXプロセス・フォーク)。親プロセスは子プロセスにネットワーク・ソケット接続を与え、アウトサイド・デーモン312の別の接続を開始するために450に戻る。454において、子プロセスは、メモリにおけるテーブルに常駐する有効な実行可能サービス（例えば、`httpsvc`）のリスト及びそれらのサービスへの完全なディレクトリ・パスでもって、そのリクエストされたサービスを有効化する。そのリクエストされたサービスが有

効サービスのリスト内にない場合、それは否定されるであろう。従って、たとえ未認証のユーザがアウトサイド・デーモン312を介してインサイド・デーモン322へのアクセスを得たとしても、そのユーザは、有効サービスのリストに常駐するサービスに限定されるであろう。

サービス・リクエストが有効である場合、455において、インサイド・デーモン322は、UNIX実行コマンドを呼び出し（即ち、それ自身を新しいサービス・プログラムでもってオーバレイし、リクエストされたサービスを実行し）、`httpsvc470`にネットワーク・ソケット接続を与える。`httpsvc470`は、アウトサイド・デーモン312の名前である1つの付加的環境変数をそのプロセス環境に加える。`SGI`は、その`SGI`が`http`デーモン314ではなく`example.pl480`を実行したことを、その`example.pl480`が必要な場合に決定できるように、追加の環境変数を加える。

側注として、アウトサイド・デーモン312、インサイド・デーモン322、`SGI`クライアント・ルーチン416、及び`httpsvc470`は、それぞれ、アカウント・ファイル及びエラー・ロギング・ファイルを有する。それぞれは、種々の情報アカウントをエラー及びアカウント・ログ内に配置させるデバッグ及びトレース・アークギュメントを有する。更に、トレース・アークギュメントが`SGI`クライアント・ルーチン416によってセットされる場

合、アウトサイド・デーモン312、インサイド・デーモン322、及び`httpsvc470`は、すべて、各々が最初に実行された時にトレーシングをセット

した方法に関係なく、それらのそれぞれのエラー・ログ・ファイルにおける特定のトランザクションをトレースするであろう。

436において、アウトサイド・デーモン312は、前に作成されたヘッダをサービス・プログラム324に送る。471において、サービス・プログラム324がそれを受け取る。それに応答して、サービス・プログラム324は、ヘッダ（オリジナル・プロセス環境変数を含む）を解析して可変値ストリングにし、`example.pl` 462において定義されたオリジナル・プロセス環境を再作成する。サービス・プログラム324は、`cgi-bin` 426においてヘッダ可変`SGICMD=example.pl`から呼び出すための適正なプログラムを決定し、`example.pl` 480とコミュニケーションするためのコミュニケーション・チャンネル（例えば、パイプ）を作成し、472において、`example.pl` 480を呼び出す。437において、アウトサイド・デーモン312は標準的な入力文字ストリーム（例えば、テキスト）をサービス・プログラム324に送る。473において、サービス・プログラム324は、そのテキストを`example.pl` 480の標準的な入力に送る。

この時点で、サービス・プログラム324は471においてオリジナル・プロセス環境を再作成した（最初は、462

において作成された）ので、`example.pl` 480は、それがSGIではなく`http`デーモン314によって472で実行されるものと見なす（任意選択的に、それは、SGIが`httpsvc` 470によってヘッダに加えられる付加的な環境変数からそれを呼び出したことを決定することができるけれども）。従って、SGIは、顧客の`http`デーモン314、及び`example.pl` 480に常駐する実際のトランザクション・プログラムの両方にとって透明である。従って、`http`デーモン314も`example.pl` 480に常駐するトランザクション・プログラムも変更される必要がない。

今や、すべての情報が、`example.pl` 481においてデータベース330上の内部トランザクションを実行するために存在する。一旦トランザクションが終了してしまうと（それが成功しても、或いは成功しなくても）、481に

において、そのトランザクションからの出力が顧客に戻される。482において、

`xample.pl` 480はそのトランザクションからの出力を受け、それをサービス・プログラム324のパイプ474に送る。474において、サービス・プログラム324は出力をアウトサイド・デーモン312に送る。438において、アウトサイド・デーモン312は出力をSGI\_\_クライアント・ルーチン416に送る。464において、SGI\_\_クライアント・ルーチン416は出力を特別パール・スクリプト`example.pl` 462に送る。

465において、`example.pl` 462は出力をデーモン314に送る。

466において、デーモン314は出力を顧客に送る。

従って、顧客が開始したトランザクションは、デーモン314からアウトサイド・デーモン312に送られ、454における検証及び481における処理のためにアウトサイド・デーモン312からインサイド・デーモン322に送られ、最後に、その出力は466において顧客に戻される。顧客リクエスト及びテキストは、すべて、SGIの完全な制御の下に、しかし、顧客にとって完全に透明に、ファイアウォール300を通して内部トランザクション処理に利用される。インサイド・デーモン322は451において認証を行い、481において任意選択的にユーザ認証を行うので、外部サーバ310の妥協案は非常に小さい内部セキュリティ・リスクをとるが、決して内部ネットワークと妥協しない。

#### 産業上の利用可能性

この特定の実施例を使用すると、既存の`http`サーバは、既存の`cgi`・ビン・コマンドに対するわずかな修正でもって又は全く修正なしでSGIを実施することができる。SGIは完全に隠蔽され、複雑な`http`サーバさえも自動的にサポートするであろう。現在の`http`サーバに対するわずかな修正でもってビジネス・トランザクションに対する更なるセキュリティ及びサポートを加えることが可能である。外

部ネットワークにとって利用可能なトランザクション (`example.pl` のようなプログラム) は、それぞれ`cgi`・ビン415及び`cgi`・ビン426に

常駐するパール・スクリプト及びトランザクション・プログラムに限定されるので、及び内部サーバ320は、通常、厳しい企業制御の下にあり、内部の開発者によって容易には修正されないので、SGIも、内部の開発者が企業の検査及び同意なしに内部トランザクションを外部の顧客にとって利用可能なものにするのを困難にしている。

本発明を、その特定の実施例に関連して示し且つ説明したけれども、本発明の精神及び技術的範囲を逸脱することなく、形式及び詳細における上記の及びその他の変更を行い得ることは当業者には明らかであろう。例えば、別の実施例は、SGI\_クライアント・ルーチン416及びアウトサイド・デーモン312をデーモン314に組み込むことが可能である。これは更に大きなパフォーマンスを提供するであろうが、http実施所有権及びそれに対する改良を組み込むことを難しくするであろう。

【図1】

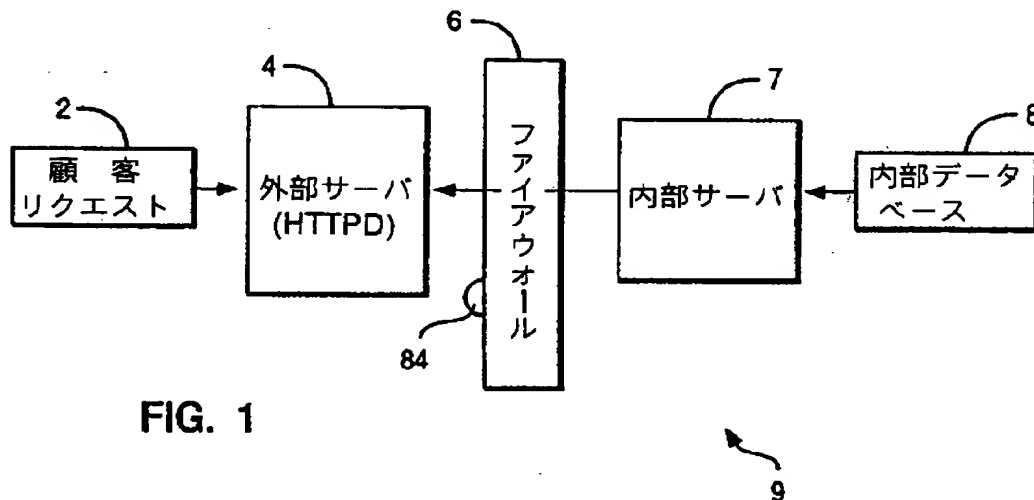


FIG. 1

【図3】

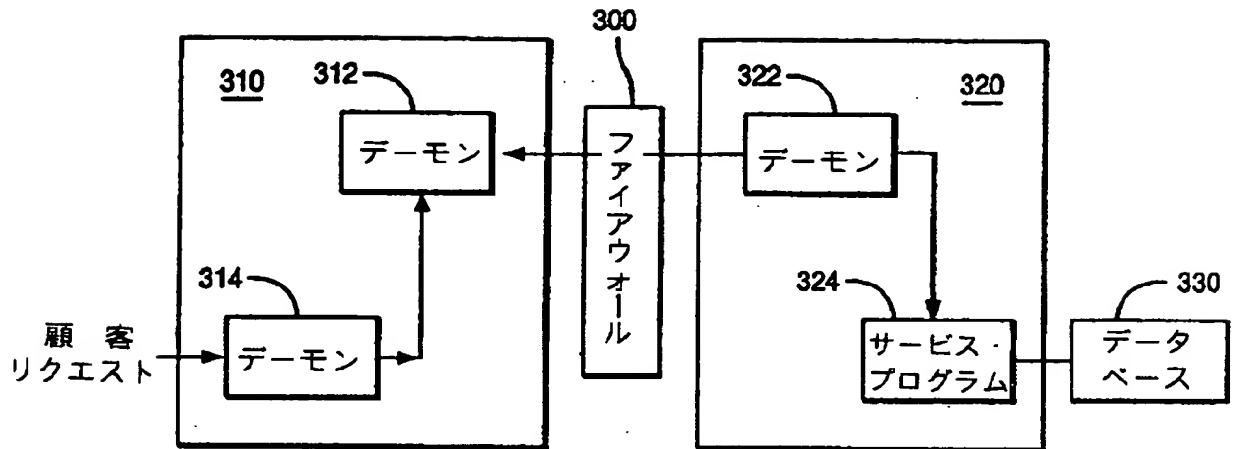


FIG. 3

【図 2】

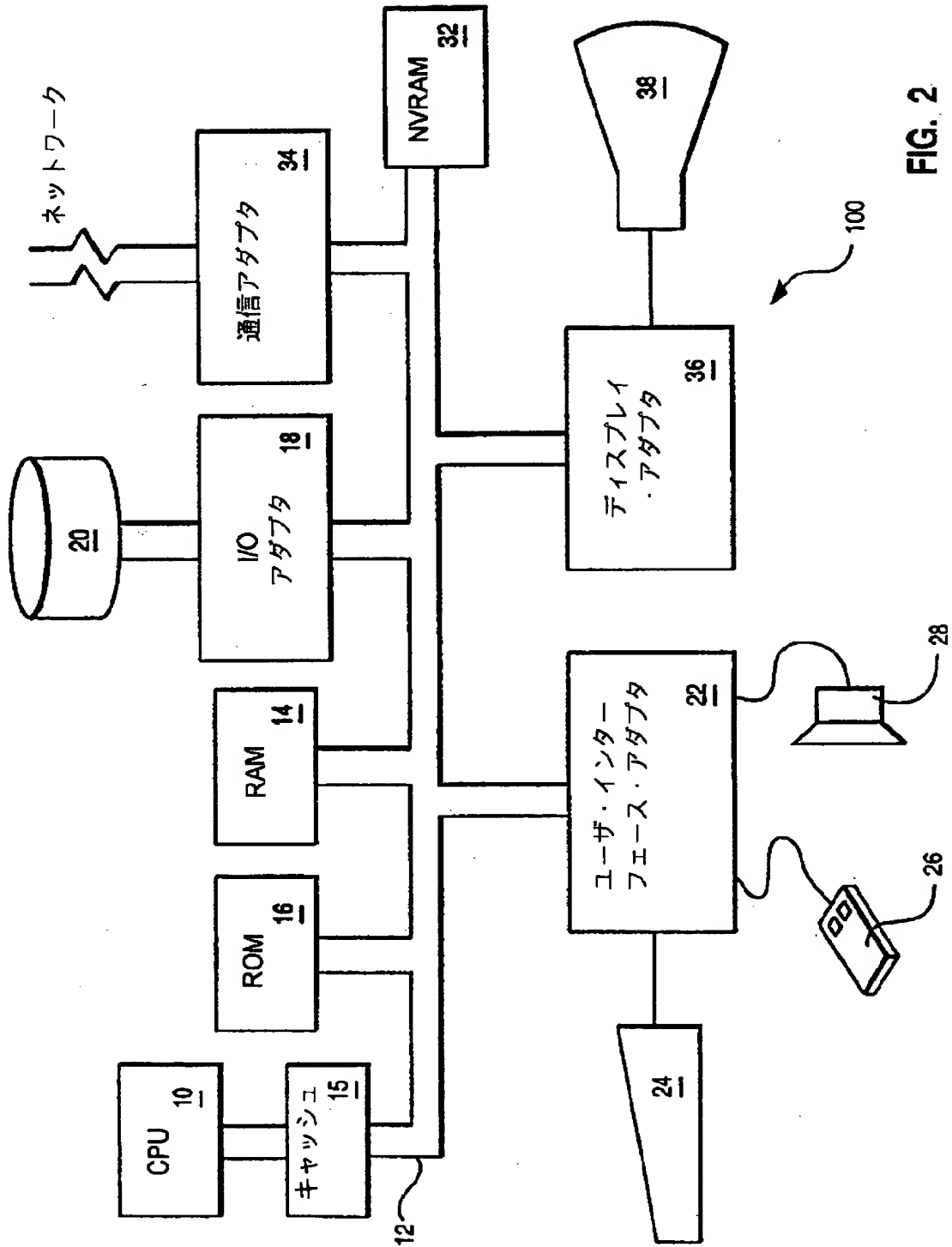
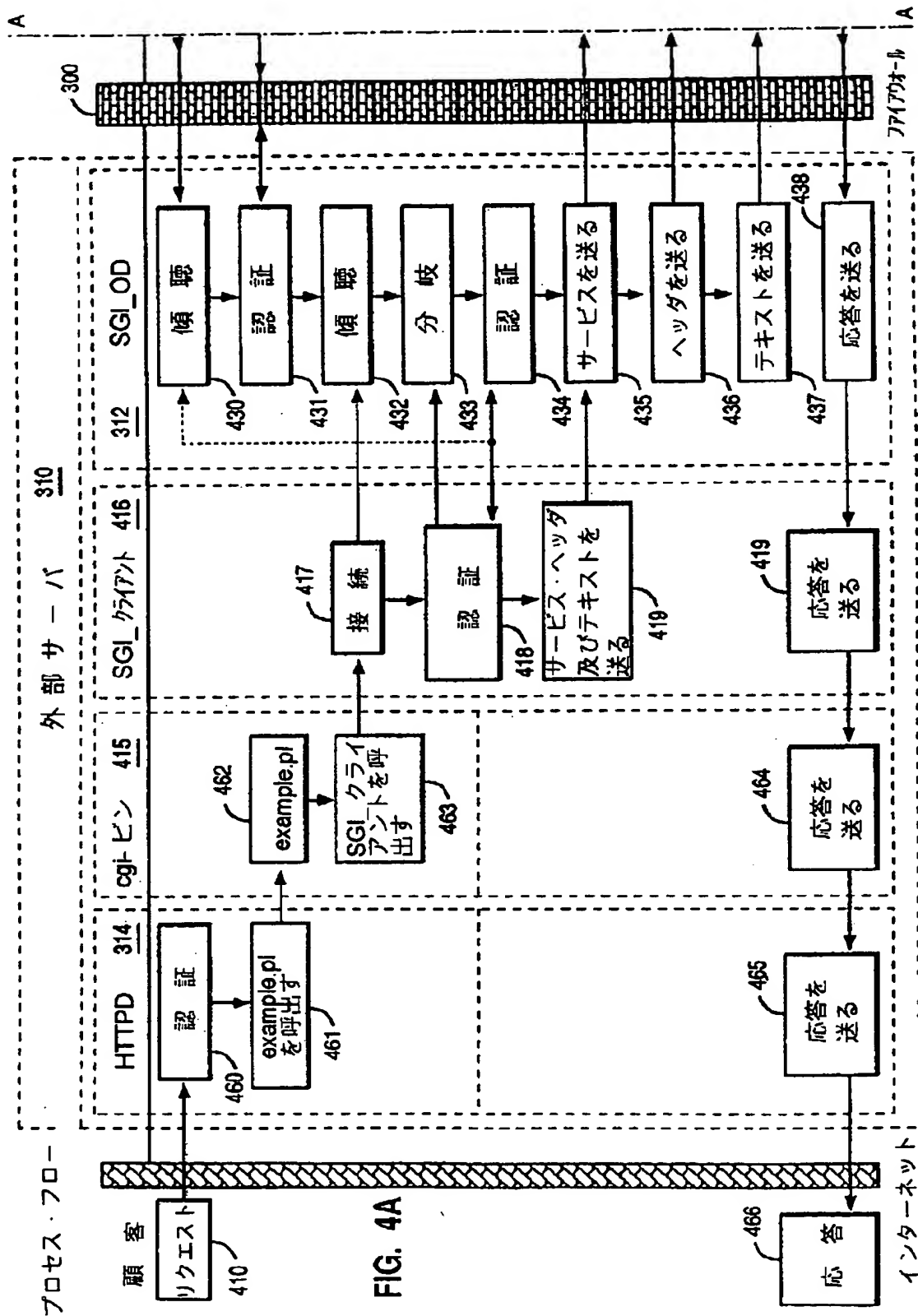


FIG. 2

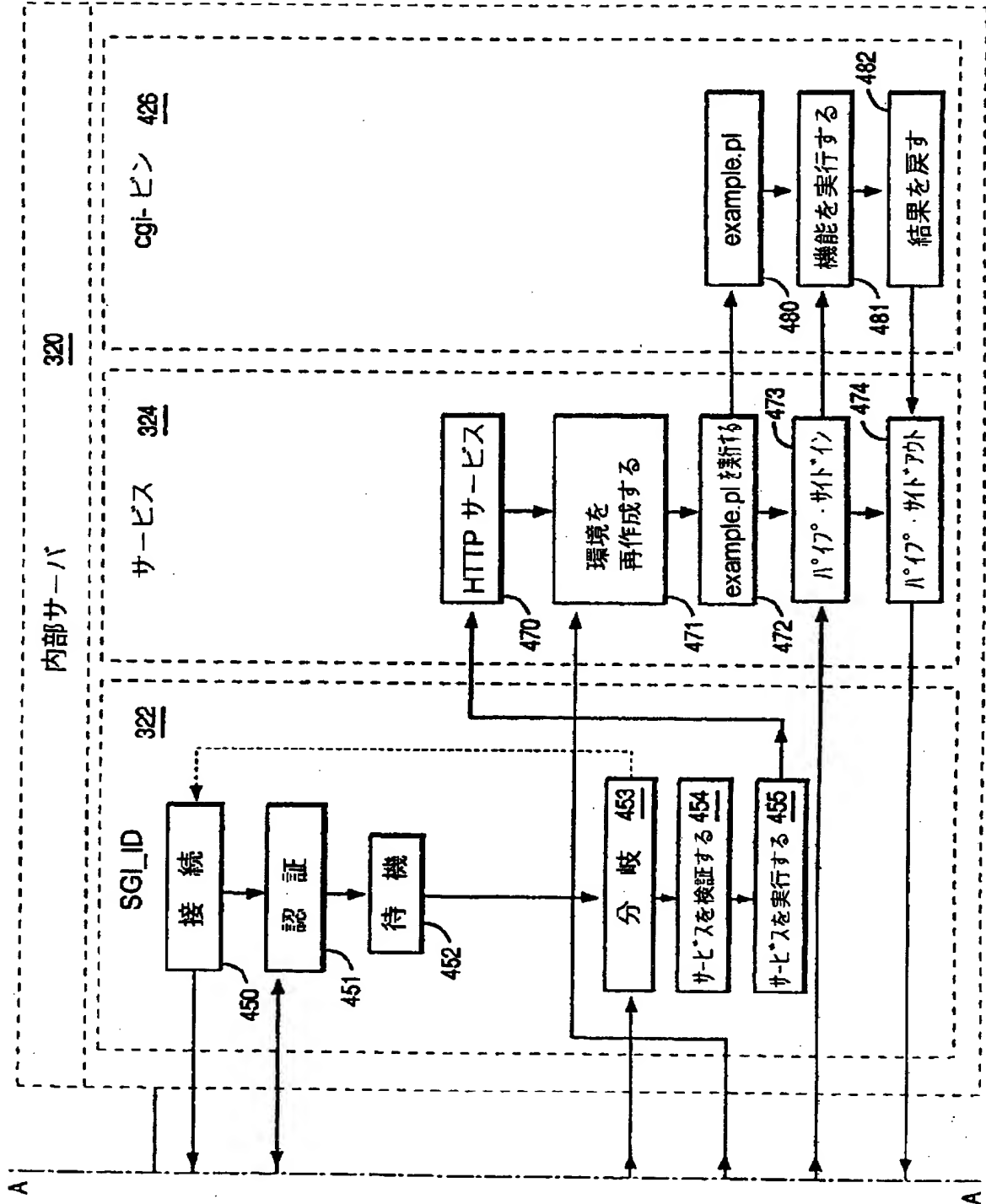


【図4】



【図4】

FIG. 4B



## 【国際調査報告】

## INTERNATIONAL SEARCH REPORT

International Application No.  
PCT/GB 95/00664

<b>A. CLASSIFICATION OF SUBJECT MATTER</b> IPC 6 H04L29/06		
According to International Patent Classification (IPC) or to both national classification and IPC		
<b>B. FIELDS SEARCHED</b> Minimum documentation searched (classification system followed by classification symbols) IPC 6 H04L		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched		
Electronic data base consulted during the international search (name of data base and, where practical, search terms used)		
<b>C. DOCUMENTS CONSIDERED TO BE RELEVANT</b>		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP,A,0 658 837 (CHECKPOINT SOFTWARE TECHNOLOGIES) 21 June 1995 see page 2, line 34 - page 3, line 5 see page 3, line 27 - page 4, line 15 see abstract	1,11,12
A	IEEE COMMUNICATIONS MAGAZINE, vol. 32, no. 9, September 1994, US, pages 50-57, XP000476555 S.M.BELLOWIN ET AL: "NETWORK FIREWALLS" see page 55, left-hand column, line 56 - page 56, left-hand column, line 54	1,11,12
<input type="checkbox"/> Further documents are listed in the continuation of box C. <input checked="" type="checkbox"/> Patent family members are listed in annex.		
* Special categories of cited documents : "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art. "&" document member of the same patent family		
Date of the actual completion of the international search		Date of mailing of the international search report
22 July 1996		07.08.96
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL - 2230 HX Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax (+31-70) 340-3016		Authorized officer Canosa Areste, C

International Application No  
PCT/GB 96/00664

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP-A-658837	21-06-95	CA-A- 2138058 JP-A- 8044642	16-06-95 16-02-96
-----			

## フロントページの続き

(51) Int. Cl. 6	識別記号	F I	
H O 4 L 12/46		H O 4 L 11/00	3 1 0 C
12/66		9/00	6 7 3 C
// G O 9 C 1/00	6 6 0		

## 【要約の続き】

リ受信されたトランザクション・リクエスト（２）を呼び出すことを含む。第３ステップは、そのトランザクション・リクエストの呼出しに応答して、外部コンピュータ・システムが、トランザクション・リクエストを実行するためにそのトランザクション・リクエスト、アークギュメント、及びプロセス環境変数を含むストリングを作成することを含む。第４ステップは、外部コンピュータ・システムが承認済みの接続を通して内部コンピュータ・システムにそのストリングを送ることを含む。第５ステップは、内部コンピュータ・システムがトランザクション・リクエストを検証することを含む。第６ステップは、内部コンピュータ・システムがオリジナル・プロセス環境を再作成することを含む。最後のステップは、内部コンピュータ・システムがトランザクション・リクエストを実行し、それによって出力を発生することを含む。

**THIS PAGE BLANK (USPTO)**

【公報種別】特許法第 17 条第 1 項及び特許法第 17 条の 2 の規定による補正の掲載

【部門区分】第 6 部門第 3 区分

【発行日】平成 13 年 8 月 14 日 (2001. 8. 14)

【公表番号】特表平 10-512696

【公表日】平成 10 年 12 月 2 日 (1998. 12. 2)

【年通号数】

【出願番号】特願平 9-517127

【国際特許分類第 7 版】

G06F 13/00 351

355

15/00 310

H04L 9/32

12/28

12/46

12/66

// G09C 1/00 660

【F I】

G06F 13/00 351 Z

355

15/00 310 A

G09C 1/00 660 E

H04L 11/20 B

11/00 310 C

9/00 673 C

## 誤訳訂正書

平成13年3月12日

特許庁長官 殿


## 1. 事件の表示

平成9年 特許願 第517127号

## 2. 特許出願人

住 所 アメリカ合衆国13504、ニューメキシコ州アーモンク（番地なし）  
 名 称 インターナショナル・ビジネス・マシーニズ・コーポレーション

## 3. 代理人

住 所 東京都新宿区西新宿七丁目21番1号、新宿ロイヤルビル3階  
 山本特許事務所  
 電話 (03) 3366 6730  
 FAX (03) 3337-6736  
 氏 名 伊藤 山 本 (印)   
 (6545)

## 4. 拒絶理由通知の日付

平成12年9月4日

## 5. 訂正の対象書類名

明細書

## 6. 訂正の対象項目名

明細書全文

## 7. 訂正の内容

明細書の記載を例証のとおり訂正する。

別紙

別紙

## 明 細 書

保護されたゲートウェイ・インターフェース

## 技術分野

本発明は、ネットワークのための保護されたゲートウェイ・インターフェースに関するものであり、更に詳しく、しかし、既定的でなく云えば、外部のネットワーク・ユーザが、セキュリティ・ファイアウォールに違反することなく内部資源を利用して、許可されたトランザクションを開始することを可能にするための保護されたゲートウェイ・インターフェースに関するものである。

## 背景技術

第1図は、従来技術の保護されたゲートウェイ・インターフェース（SGI）9を示す。そのSGIは、外部ネットワーク（例えば、インターネット）上の誰かからのユーザ/顧客リクエスト2を処理するための外部サーバ4（例えば、ハイパーテキスト転送プロトコル・デモンストラータ（HTTPD））及び内部ネットワーク上の誰か（例えば、特定の企業のために働く誰か）及び内部データベース8からのリクエストを処理するための内部サーバ7を有する。SGI 9は、更に、内部データベース8における内部トランザクションを外部から開始させることを防ぐためのファイアウォール6を含む。従っ

て、ファイアウォール6は、外部の顧客が内部ネットワーク（即ち、内部サーバ7及び内部データベース8）への直接接続を開始することを防ぐ。この制限は、顧客が外部ネットワークから内部トランザクションを開始には開始させることができないので、製品及びサービスの購入リクエストのような有効なトランザクションも禁止してしまう。

上記の問題点に対する一時的な解決方法は、インバウンド・トラフィックに対する特定のポート（例えば、ポート84）をファイアウォール6において開くことを必要とする。しかし、この解決方法は、明らかに、内部ネットワークを外部の攻撃を免れ得ない状態にしておくことになる。更の解決方法は、すべての必要な資源（例えば、データベース8）を外部サーバ4上に設置することである。しかし、この解決方法は内部トランザクションの実行を禁止したままである。更に、外部サーバ4はすべての必要な資源を保持するに十分な記憶装置を持たないことがあり、或いは、それらの資源は秘密性が高くて外部サーバ上に配置することができず、提供可能なサービスを制限することがある。

従って、セキュリティ・ファイアウォールに違反することなく内部ビジネス資源を利用して顧客が許可されたビジネス・トランザクションを開始することを可能にするための技法に対する大きな需要がある。

発明の開示



別紙

別紙

従って、コンピュータ読取り可能なプログラム手段を具体化するコンピュータ実施方法、独自にプログラムされたコンピュータ・システム、及びその生産物は、外部ネットワークにおける顧客が、セキュリティ・ファイアウォールに違反することなく内部ネットワーク上の内部ビジネス資源を利用して、許可されたビジネス・トランザクションを開始することを可能にする。

具体的に云えば、その方法は、外部コンピュータ・システムが、内部コンピュータ・システムとその外部コンピュータ・システムとの間のセキュリティ・ファイアウォールに違反することなく内部資源を使用してトランザクション・リクエストを開始することを可能にするように内部コンピュータ・システムに指示する。その方法は、内部コンピュータ・システムと外部コンピュータ・システムとの間の内部コンピュータ・システムによって開始される接続を認証し、それによって認証済み接続を確立する第1ステップを含む。第2ステップは、外部コンピュータ・システムがその外部コンピュータ・システムによって受信されたトランザクション・リクエストを呼び出すことを含む。第3ステップは、トランザクション・リクエストの呼出に応答して、外部コンピュータ・システムがそのトランザクション・リクエストとそのトランザクション・リクエストを実行するためのプロセス環境を生成を含むストリングを作成することを含む。第4ステップは、外部コンピュータ・システムが、そのストリングを、認証済み

接続を通して内部コンピュータ・システムに送ることを含む。第5ステップは、内部コンピュータ・システムがそのトランザクション・リクエストを検証することを含む。第6ステップは、内部コンピュータ・システムがオリジナル・プロセス環境を再作成することを含む。最後のステップは、内部コンピュータ・システムがそのトランザクション・リクエストを実行し、それによって出力を発生することを含む。

従って、本発明の目的は、ユーザ及び環境のトランザクション・プログラムにとって透明である保護されたゲートウェイ・インターフェースを作成することにある。

本発明のもう一つの目的は、ユーザが、内部資源を利用して、内部ネットワークに対するファイアウォールを通してトランザクションを有効に開始することを可能にすることにある。

本発明の更にもう一つの目的は、ユーザが、認証されたトランザクションの有効なセットのみを開始することを可能にすることにある。

本発明の更にもう一つの目的は、外部コンピュータ・システムがユーザからのトランザクション・リクエストを受け取る前に、内部コンピュータ・システムと外部コンピュータ・システムとの間の接続を安全に承認することにある。

本発明の更にもう一つの目的は、トランザクション・プログラムを修正する必要なくそれらをファイアウォール内に記憶することにある。

別紙

別紙

#### 図面の簡単な説明

添付図面を参照して、本発明を実施例によって説明することにする。

第1図は、本発明を実施する場合に使用するための通常のネットワーク・システムのブロック図を示す。

第2図は、本発明を実施するための代表的なハードウェア構成を示す。

第3図は、好適な実施例による保護されたゲートウェイ・インターフェース(SGI)のブロック図を示す。

第4図は、第3図に示されたSGIの更に詳細なプロセス・フロー図を示す。

#### 発明を実施するための最良の形態

好適な実施例は、コンピュータ実施される方法、独自にプログラムされたコンピュータ・システム及びメモリを含み、それらは、外部のユーザ/顧客が、セキュリティ・ファイアウォールに違反することなく、内部ビジネス資源を利用して一つの認証済みビジネス・トランザクションを開始させることを可能にするように内部コンピュータ・システムに指示するための詳細なロジックを具体化するものである。

本発明は、第2図に示されたコンピュータ・システムにおいて実施される。コンピュータ・システム100は、キャッシュ15、ランダム・アクセス・メモリ(RAM)14、接

取り専用メモリ(ROM)16、及び不揮発性RAM(NVRAM)32を処理するためのIBM社のPowerPC601又はIntel社の486マイクロプロセッサのような中央処理装置(CPU)10を含む。I/Oアダプタ18によって制御される一つ又は複数のディスク20が長期記憶装置を提供する。テープ、CD-ROM、及びWORMドライブを含む他の種々の記憶媒体が使用可能である。又、データ又はコンピュータ・プロセス命令を記憶するための取り外し可能な記憶媒体も具備可能である。(なお、IBM及びPowerPCは国際的な標準であり、Intelはインテル・コーポレーションの商標である)。

Sun Solaris、マイクロソフト社のWindows NT、IBM社のOS/2、或いはアップル社のSystem 7のような任意の適当なオペレーティング・システムがデスクトップからの命令及びデータがRAM14からCPU10を制御する。従って、そのデスクトップはRAM14から実行する。しかし、好適な実施例では、IBM RISC System/6000が、UNIXオペレーティング・システムのIBM社の実施形態であるAIXオペレーティング・システムを走らせる。しかし、前述のように、本発明を実施するために、他のハードウェア・プラットフォーム及びオペレーティング・システムを利用することが可能であることは当業者には容易に明らかであろう。(なお、Solaris

図解

risはサン・マイクロシステムズ社の商標であり、System 7はアップル・コンピュータ社の商標であり、O/S 2、RISC System/6000、及びAIXはIBM社の商標である。UNIXは、X/Open社を通して排他的に使用許諾された米国及びその他の国における商標である。

ユーザは、ユーザ・インターフェース・アダプタ22によって制御されるI/O装置（即ち、ユーザ・コントロール装置）を通してコンピュータ・システム100とコミュニケーションする。ディスプレイ38はユーザに情報を表示し、一方、キーボード24、ポインティング装置26、及びスピーカ28はユーザがコンピュータ・システムに指示を行うことを可能にする。通信アダプタ34は、このコンピュータ・システムとネットワークに接続された他の処理装置との間の通信を制御する。ディスプレイ・アダプタ36は、このコンピュータ・システムとディスプレイ38との間のコミュニケーションを制御する。

第3図は、好適な実施例による保護されたゲートウェイ・インターフェース（SGI）のブロック図及びプロセス・フローを示す。SGIは一对のサーバ310及び320に常駐し、そのサーバの各々はコンピュータ・システム100（第1図参照）において実現される。外部サーバ310はファイアウォール300の外側に常駐し、一方、内部サーバ320はファイアウォール300の内側に常駐する。ファイアウォール300は、外部トランザクションがそれを通して内部サーバ320に達するのを防ぐ任意の適当な一般的なファイアウォールを使用して実現される。好適な実施例では、ファイアウォール300はネットワーク・ルータ（例えば、シスコ・ルータ）である。しかし、ファイアウォール300が内部サーバ320内に常駐してもよいことは当業者には容易に明らかであろう。

外部サーバ310は、インターネットのような外部のネットワークにおけるユーザ/顧客とのコミュニケーションを管理する。しかし、任意の公衆網又は専用網におけるSNA又はX.25のような任意のタイプの通信プロトコルが使用可能であることは当業者には明らかであろう。内部サーバ320は、内部コーポレート情報ネットワークのような内部ネットワーク上の内部資源（例えば、データベース330）のコミュニケーションを管理する。外部サーバ310は、アウトサイド・デーモン312を走らせ、一方、内部サーバ320はインサイド・デーモン322を走らせ、それによって、ファイアウォール300にまたがるコミュニケーションを可能にする。デーモンは、外部事象を待ち、それらの事象が生じる時にいつも事前定義された一連のアクションを実行する長時間実行コンピュータ・プログラムである。デーモンはサービス・リクエストを傾聴し、リクエストされた時にそれらを実行する。外部サーバ310は、外部ネットワークからのサービス・リクエストを傾聴するデーモン314も走らせる。内

図解

図解

部サーバ320は、所望の内部トランザクションを実行するためのサービス・プログラム324を含む。サービス・プログラム324及び内部データベース330は、ビジネス・トランザクション（以下で、更に詳細に説明される）を実施する一組のコンピュータ・プログラムを表す。

第4A図及び第4B図は、前に第3図に示したSGIの更に詳細なプロセス・フローを示す図である。外部サーバ310は、任意の適当な通常の通信プロトコル・デーモン（HTTTPD）314、cgi\_ビン416、SGI\_クライアント・ルーチン416、及びアウトサイド・デーモン（SGI\_OD）312を含む。アウトサイド・デーモン312は、SGI\_クライアント・ルーチン416及びインサイド・デーモン（SGI\_ID）322とコミュニケーションするためのクライアント/サーバ・ソフトウェアを含む。SGI\_クライアント・ルーチン416は、アウトサイド・デーモン312とコミュニケーションするためのクライアント/サーバ・ソフトウェアを含む。cgi\_ビン416は、デーモン314によって実行されるソフトウェアのディレクトリである。特に、この例では、cgi\_ビンは、SGI\_クライアント・ルーチン416とコミュニケーションするための特別バッチ・スクリプト（以下で、更に詳細に説明される）であるexample.pl 462を含む。好適な実施例では、デーモン314は、通常のハイパーテキスト転送プロトコル・デーモン（HTTTPD）（一般には、ウェブ・サーバとしても知られてい

図解

る）である。

内部サーバ320は、インサイド・デーモン322、サービス・プログラム324、及びcgi\_ビン426を含む。サービス・プログラム324は、アウトサイド・デーモン312、インサイド・デーモン322とコミュニケーションし、cgi\_ビン・ルーチン（例えば、example.pl 480）を実行する。この例では、example.pl 480は、ユーザ/顧客を承認するために及びビジネス・トランザクションを実行するために、内部コーポレート・データベース（例えば、第3図におけるコーポレート・データベース330）とコミュニケーションする。

顧客/ユーザが410においてトランザクションをうまくリクエストし得る前に、内部サーバ320及び外部サーバ310は適正に接続しなければならない。それを行うために、外部オペレーティング・システムはアウトサイド・デーモン312を実行して、外部サーバ310におけるファイルシステム（図がされてない）に存在するパスワード・ファイルのコミュニケーション・ポート及びロケーションを識別する。一方、アウトサイド・デーモン312はパスワード・ファイルから8文字パスワードを読み取り、その識別されたポートにおいてソケットを作成し、そのソケットにおいてインサイド・デーモン322からの接続コールを傾聴する。従って、アウトサイド・デーモン312はサーバの役割を負い、430において、クライアントの役割を負ったインサイド・デー

別紙

モン322からの接続コールを待つ。更に、アウトサイド・デーモン312は第2ポートにおいてソケットを作成し（デーモン312のコミュニケーション・ポート+1）、432においてSGT\_クライアント・ルーティン416からの接続試行（以下で、更に詳細に説明される）を待つ。

内部オペレーティング・システムはインサイド・デーモン322を実行して、インサイド・デーモン322をアウトサイド・デーモン312に接続するためのコミュニケーション・ポート、外部サーバ310のホスト名、内部サーバ320におけるファイル・システム（図示されてない）に常駐するパスワード・ファイルのロケーション、及び内部サーバ320におけるファイル・システム（図示されてない）に常駐する有効サービス・ファイルのロケーションを識別する。一方、インサイド・デーモン322はパスワード・ファイルから8文字パスワードを読み取り、サービス・ファイルを読み取って有効サービスのテーブルをメモリに記憶し、識別されたコミュニケーション・ポートにおいてソケットを作成し、最後に、450において、430で接続しているアウトサイド・デーモン312にファイアウォール300を越えて標準的な接続コールを発生する。その接続は内部サーバから開始されようとするので、ファイアウォール300はその接続を許可する。

インサイド・デーモン322及びアウトサイド・デーモン312が接続した後、インサイド・デーモン322及びアウ

別紙

```
int i;

bzero (buf, sizeof(buf)); /* バッファをクリアする */
strcpy (buf, cred); /* パスワードをバッファにロードする */

/* バッファの各文字をマングルする */

for (i = 0; i < 8; i++) {
    buf[i] ^= (auth_time & 0177); /* タイムスタンプを論理的にANDし、結果を各文字と排他的にORする；各反復時にタイムスタンプを修正する */
    auth_time >>= 4; /* タイムスタンプをビット単位移動する */
}

for (i = 0; i < 8; i++)
    if (buf[i] == 0) /* 有効文字ストリングはヌルを含むことができないので、すべてのヌルを1に置き換える */
        buf[i] = 1;
```

別紙

トサイド・デーモン312は相互に適正に認証しなければならない。それを行うために、インサイド・デーモン322は、現在のタイムスタンプを検索するために内部オペレーティング・システムへのコールを開始し、そのタイムスタンプをアウトサイド・デーモン312に送り、それに応答して認証ストリングを待つ。アウトサイド・デーモン312はそのタイムスタンプを受け取り、そしてインサイド・デーモン322によって与えられたタイムスタンプでもってその8文字パスワードをマングルすること（後述の変更すること）によって認証ストリングを作成し、このマングルされた文字ストリングを標準的なUNIXクリプト・コマンド（又は、DESのような任意の適当な暗号化アルゴリズム）でもって暗号化し、そして、431において、その結果生じた認証ストリングをインサイド・デーモン322に送る。下記のC言語は、8文字パスワードをタイムスタンプでもってマングルするプロセスを示す。この“create\_auth”コードは3つの引数が必要とする。それらの引数の第1はタイムスタンプ（即ち、auth\_time）であり、その第2はパスワード（即ち、“cred”）であり、その第3は発生された認証ストリングを記憶するためのバッファである。

```
int create_auth (time_t, char * cred * p)
{
    char buf[9]; /* 一時的バッファ */
```

別紙

```
    strcpy (p, crypt (buf, "aa") + 2); /* キーの代わりに98を使用してバッファを暗号化する */
    /* 暗号化結果のうちの（キーaaである）最初の2文字をスキップする */
    /* 暗号化結果を、Pによって指定されたユーザ供給バッファにコピーする */

    return 0;
}
```

インサイド・デーモン322が同様にそのパスワードをタイムスタンプでもってマングルし、それを暗号化し、そしてそれをアウトサイド・デーモン312によって与えられた認証ストリングと比較する。それらの認証ストリングが一致する時、プロセスは反転され、アウトサイド・デーモン312が同様にインサイド・デーモン322を認証する（即ち、外部のオペレーティング・システムから新しいタイムスタンプを得て、そのタイムスタンプをインサイド・デーモン322に送り、インサイド・デーモン322はそのパスワードをその新しいタイムスタンプでもってマングルし、それを暗号化し、そして有効化するためにそれをアウトサイド・デ

別紙

別紙

…モン312に戻す)。

この認証プロセスは、アウトサイド・デーモン312及びインサイド・デーモン322が共に知っている8文字パスワード、タイムスタンプによってランダム化された文字マングリング機能、及び暗号化プロセスを使用する。マングリング機能のために、上記のプロセスは、各認証及びすべてのトランザクションに対して異なる暗号化された認証ストリングを生じる。これは、捕捉された認証ストリングがその後の如何なるトランザクションに対しても価値がないので、攻撃に対するその脆弱性をかなり減じる。

インサイド・デーモン322及びアウトサイド・デーモン312が相互に認証した後、前にクライアントとして作用したインサイド・デーモン322は今やサーバの役割を自負、452において、アウトサイド・デーモン312が453でサービス・ストリングを供給するのを待つ。アウトサイド・デーモン312は第2の指定されたポートにおいてもう1つのソケットを作成し、432においてSGI\_クライアント・ルーチン416からの接続試行を待つ(超える)。従って、アウトサイド・デーモン312は、インサイド・デーモン322に関する類似クライアント(情報はそれらの間で送られる)及びSGI\_クライアント・ルーチン416に関するサーバという2つの役割を食う。

デーモン314は、今や、顧客リクエスト410を受け付けるように準備される。顧客リクエストは、例えば、特定の

株式又は金融市場に関する調査情報を購入するためのトランザクションであってもよい。410において、顧客は、httpクライアント・アプリケーション・ユーザ・インターフェースを走っている顧客のシステムにおける特定のアイコン又は強調表示されたフェーズ上でクリックすることにより、次のようなトランザクション・リクエストを実行することを決定する。

`http://external_server/cgi-bin/example.pl?stock1+stock2`

そのhttpクライアント・ユーザ・インターフェースは、一般に、詳細なトランザクション情報(例えば、株式又は金融市場情報)及び現金情報(例えば、クレジット・カード番号)をユーザに尋ねる。ユーザは、リクエストされたサービスが**認証されたユーザのみに与えられる場合**、そのユーザのユーザid及びパスワードを入力するように要求されることもある。

送られたユーザ入力のフォーマットは、トランザクションを実施するために使用されたハイパ・テキスト・マークアップ言語(HTML)フォームのタイプに依存する。2つのタイプの一般的なHTMLフォームが存在する。「GET」タイプはコマンド・ライン上にすべてのユーザ入力を配置する。従って、株式1(stock1)、株式2(stock2)、及び他の任意のユーザ入力下記のようにコマンド・ラインの一部となるであろう。

別紙

別紙

`.../cgi-bin/example.pl?stock1+stock2+chargecardnumber+expdate`

しかし、コマンド・ラインはネットワークを通してクリア・テキストとして送られるので、顧客のクレジット・カード番号及び有効期限日をネットワークを通して送ることは適切ではない。従って、暗号化を伴うHTMLの「PUT」タイプは、クレジット・カード番号及び有効期限日がネットワークを通して安全に送られるように使用される。この情報をすべて供給した後、httpクライアント・アプリケーションは、410において、httpを介して外部サーバ310にリクエストを送る。

460では、デーモン314が、一般に知られそして導入されているHTTP認証技法(例えば、顧客のパスワードを標準的なUNIXクリプト・コマンドでもって暗号化し、その結果を、デーモン314に常駐するhttpパスワード・ファイルにおけるパスワード・エントリと比較する)に従って顧客のパスワードを認証する。ユーザid及びパスワードが有効である場合、461において、デーモン314は「PUT」フォームを認識し、文字ストリームを自動的に非暗号化し、適正なUNIXプロセス環境を作成する。デーモン314は、PATH、USERNAME、LOGNAME、及びAUTHTYPE変数を含む標準的なUNIXプロセス環境を作成するための一般に知られた通常のhttp構成ファイル(図示されてない)を含む。その後、httpsvcl

70は(後述の)471においてこのプロセス環境を再作成する。一旦、プロセス環境が作成されてしまうと、デーモン314は、`example.pl 462 (cgi-ビン415に常駐しなければならない)`を実行して**任意の必要な引数(例えば、stock1及びstock2)をそれに送り、ユーザ入力をexample.pl 462の標準的な入力ストリームに送る。**

`example.pl 462`がcgi-ビン415に常駐すると仮定すると、**ファイアウォール300が存在しなければexample.pl 462は内部データベース330(第3図参照)と直接にコミュニケーションし、所望のトランザクションを遂行するであろう。**しかし、ファイアウォール300が存在し、`example.pl 462`が内部データベース330と直接にコミュニケーションしないので、`example.pl 462`は実際のトランザクション・プログラムではない。むしろ、実際にトランザクション・プログラムは、ファイアウォール300の内側にある`example.pl 480`としてcgi-ビン426に常駐する。従って、cgi-ビン415は、cgi-ビン426に常駐する実際のトランザクション・プログラムを実行する同じコマンドを使用して実行される「特別」パル・スクリプト(例えば、`example.pl 462`)を含む。別の方法として、各サービスが同じ方法でSGI\_クライアント・ルーチン416を呼び出すために「特別」パル・スクリプトを必要と

別紙

する多くのサービスを、外部サーバ310が提供する時、`example.pl 462`は、`cgi-bin 415`に常駐する単一のパール・スクリプトに対する像徴的なリンク（即ち、間接的ファイル参照）となり得る。更に重大なことに、顧客にとって利用可能なリクエストは、それぞれ`cgi-bin 415`及び`cgi-bin 426`に常駐するパール・スクリプト及び対応するトランザクション・プログラムに限定される。

パール・スクリプト`example.pl 462`は、デモン314から送られたすべての引数をプロセス環境に配置し（例えば、`SGIARG1=stock1`;`SGIARG2=stock2`）、その名前（その名前によってそれが呼び出される。この場合、`example.pl`）をプロセス環境に配置し（例えば、`SGICMD=example.pl`）、`UNIX env`コマンド（プロセス環境変数をデンプする）を実行し、最後に、すべてのプロセス環境変数をヘッダ・ストリングに配置する。今や、ヘッダ・ストリングは、例えば、以下のように見える。

```
"PATH=/bin:/usr/bin\nAUTHTYPE=PEM\nUSERNAME=JohnDoe\nSGIARG1=stock1\nSGIARG2=stock2\nSGICMD=example.pl")
```

次に、463において、パール・スクリプト`example.pl 462`は、外部オペレーティング・システムを呼び

別紙

出してその指定された第2ポート（デモン312のコミュニケーション・ポート+1）を検索させ、`SGI_クライアント・ルーチン416`を実行し、リクエストされたサービスのタイプ（例えば、`httpsvc`）、指定された第2ポート、外部サーバのホスト名、ヘッダ・ストリング、及び顧客のユーザidを送る。又、`example.pl 462`は、任意の標準的な入力文字ストリーム（例えば、ユーザ入力テキスト）を`SGI_クライアント・ルーチン416`に標準的な入力として送る。その後、`example.pl 462`は、469において、`SGI_クライアント・ルーチン416`から受け取ったすべての出力をデモン314に送るであろう。

`SGI_クライアント・ルーチン416`が463においてそれに送られた情報を使用して実行する時、`SGI_クライアント・ルーチン416`はアウトサイド・デモン312への認証された接続を確立する。それを行うために、417において、`SGI_クライアント・ルーチン416`は、外部サーバ310に常駐する専用のクライアント・パスワード・ファイル（図示されてない）から8文字パスワードを読み取り、432において第2ソケット接続から確立しているアウトサイド・デモン312への接続を、指定された第2ポートにおいて確立する。433において、アウトサイド・デモン312はそれ自身のコピーを作成し、それを実行する（例えば、`UNIX`プロセス・フォーク）。親プロセスは子プロセ

別紙

スにソケット接続を与え、430に戻ってインサイド・デモン322からの別のコールを待つ。

434において、子プロセスは`SGI_クライアント・ルーチン416`を認証する。それを行うために、アウトサイド・デモン312もまた、外部サーバ310に常駐する専用クライアント・パスワード・ファイル（図示されてない）から8文字パスワードを読み取る。アウトサイド・デモン312は、現在のタイムスタンプを検索するために外部オペレーティング・システムへのコールを開始し、432においてそのタイムスタンプを`SGI_クライアント・ルーチン416`に送り、それに応答して認証ストリングを待つ。`SGI_クライアント・ルーチン416`はそのタイムスタンプを受け取り、アウトサイド・デモン312によって供給されたタイムスタンプをもってその8文字パスワードをマングルすることによって認証ストリングを作成し、このマングルされた文字ストリングを標準的な`UNIX`暗号コマンドをもって暗号化し、しかる後、434において、その結果生じた認証ストリングをアウトサイド・デモン312に送る。アウトサイド・デモン312は、同様にそのパスワードをタイムスタンプをもってマングルし、それを暗号化し、それを`SGI_クライアント・ルーチン416`によって供給された認証ストリングと比較する。それらの認証ストリングが一致する場合、`SGI_クライアント・ルーチン416`は認証される。

別紙

419において、認証が成功する場合、`SGI_クライアント・ルーチン416`は、リクエストされたサービスのタイプをアウトサイド・デモン312に送る。この例では、`SGI_クライアント・ルーチン416`は、そのルーチン416が`HTTP`デモン314によって間接的に呼び出されるので、いつも`HTTP`サービスをリクエストする。特別のパール・スクリプト（即ち、`example.pl 462`）は、リクエストされたサービスが`httpsvc`であることを表す引数を使用して`SGI_クライアント・ルーチン416`を前に実行した。一方、アウトサイド・デモン312は、435において、インサイド・デモン322に「`httpsvc`」サービス・リクエストを送る。

452において、インサイド・デモン322は、アウトサイド・デモン312からのサービス・リクエストを受け取り、インサイド・デモン322は、アウトサイド・デモン312からのサービス・リクエストを受け取り、それ自身の複写イメージを作成し、それを実行する（例えば、`UNIX`プロセス・フォーク）。親プロセスは子プロセスにネットワーク・ソケット接続を与え、アウトサイド・デモン312への別の接続を開始するために450に戻る。454において、子プロセスは、メモリにおけるテーブルに常駐する有効な実行可能サービス（例えば、`httpsvc`）のリスト及びそれらのサービスへの完全なディレクトリ・パスをもって、そのリクエストされた

別紙

別紙

サービスを有効化する。そのリクエストされたサービスが有効サービスのリスト内にない場合、それは否定されるであろう。従って、たとえ未認証のユーザがアウトサイド・デーモン312を介してインサイド・デーモン322へのアクセスを得たとしても、そのユーザは、有効サービスのリストに常駐するサービスに限定されるであろう。

サービス・リクエストが有効である場合、455において、インサイド・デーモン322は、UNIX実行コマンドを呼び出し（即ち、それ自身を新しいサービス・プログラムでもってオーバレイし、リクエストされたサービスを実行し）、`httpsvc470`にネットワーク・ソケット接続を与える。`httpsvc470`は、アウトサイド・デーモン312の名前である1つの追加的環境変数をそのプロセス環境に加える。SGIは、そのSGIがexample.pl 480を実行したのであって、httpデーモン314が実行したのではないということ、必要な場合に、`example.pl 480`が決定できるように、その追加の環境変数を加える。

側注として、アウトサイド・デーモン312、インサイド・デーモン322、SGIクライアント・ルーチン416、及び`httpsvc470`は、それぞれ、アカウンティング・ファイル及びエラー・ロギング・ファイルを有する。それぞれは、異なる量の情報をエラー及びアカウンティング・ログ内に配置させるデバッグ及びトレース引数を有する。

別紙

別紙

この時点で、サービス・プログラム324は471においてオリジナル・プロセス環境（最初に、462において作成された）を再作成したので、`example.pl 480`は、それがSGIではなくhttpデーモン314によって472で実行されるものと見なす（任意選択的に、それは、SGIが`httpsvc470`によってヘッダに加えられる追加の環境変数からそれを呼び出したことを決定することができる）。従って、SGIは、顧客のhttpデーモン314、及び`example.pl 480`に常駐する実際のトランザクション・プログラムの両方にとって透明である。従って、httpデーモン314も`example.pl 480`に常駐するトランザクション・プログラムも変更される必要がない。

今や、すべての情報が、481において`example.pl 480`がデータベース330上の内部トランザクションを実行するために存在する。一旦トランザクションが終了してしまうと（それが成功しても、或いは成功しなくても）、481において、そのトランザクションからの出力が顧客に戻される。482において、`example.pl 480`はそのトランザクションからの出力を受け、それをサービス・プログラム324のパイプ474に送る。474において、サービス・プログラム324は出力をアウトサイド・デーモン312に送る。438において、アウトサイド・デーモン312は出力をSGIクライアント・ルーチン416に送る。

更に、トレース引数がSGIクライアント・ルーチン416によってセットされる場合、アウトサイド・デーモン312、インサイド・デーモン322、及び`httpsvc470`は、すべて、各々が最初に実行された時にトレーシングをセットした方法に関係なく、それらのそれぞれのエラー・ログ・ファイルにおける特定のトランザクションをトレースするのである。

436において、アウトサイド・デーモン312は、前に作成されたヘッダをサービス・プログラム324に送る。471において、サービス・プログラム324がそれを受け取る。それに応答して、サービス・プログラム324は、ヘッダ（オリジナル・プロセス環境変数を含む）を解析して可変値ストリングにし、`example.pl 482`において定義されたオリジナル・プロセス環境を再作成する。サービス・プログラム324は、`cgi-bin426`においてヘッダ可変SGICMD=`example.pl`から呼び出すための適正なプログラムを決定し、`example.pl 480`とコミュニケーションするためのコミュニケーション・チャネル（例えば、パイプ）を作成し、472において、`example.pl 480`を呼び出す。437において、アウトサイド・デーモン312は標準的な入力文字ストリーム（例えば、テキスト）をサービス・プログラム324に送る。473において、サービス・プログラム324は、そのテキストを`example.pl 480`の標準的な入力に送る。

464において、SGIクライアント・ルーチン416は出力を特別バブル・スクリプト`example.pl 462`に送る。465において、`example.pl 462`は出力をデーモン314に送る。466において、デーモン314は出力を顧客に送る。

従って、顧客が開始したトランザクションは、デーモン314からアウトサイド・デーモン312に送られ、454における検証及び481における処理のためにアウトサイド・デーモン312からインサイド・デーモン322に送られ、最後に、その出力は466において顧客に戻される。顧客リクエスト及びテキストは、すべて、SGIの完全な制御の下に、しかし、顧客にとって完全に透明に、ファイアウォール300を通して内部トランザクション処理に利用される。インサイド・デーモン322は451において検証を行い、厳密には454において外部ネットワークによって利用可能なサービスを強化し、481において任意選択的にユーザ認証を行うので、外部サーバ310という妥協点は非常に小さい内部セキュリティ・リスクしか提示せず、決して内部ネットワークに侵害を及ぼさせるものではない。

#### 産業上の利用可能性

この特定の実施例を使用すると、既存のhttpサーバは、既存のcgi-bin・コマンドに対するわずかな修正でもって又は全く修正なしでSGIを実施することができる。SG

別紙

Iは完全に隠蔽され、複雑なhttpサーバさえも自動的にサポートするであろう。現在のhttpサーバに対するわずかな修正でもってビジネス・トランザクションに対する更なるセキュリティ及びサポートを加えることが可能である。外部ネットワークにとって利用可能なトランザクション(example.plのようなプログラム)は、それぞれcgi-bin415及びcgi-bin426に常駐するパール・スクリプト及びトランザクション・プログラムに限定されるので、及び内部サーバ320は、通常、厳しい企業制御の下にあり、内部の開発者によって容易には修正されないで、SGIも、内部の開発者が企業の検査及び同意なしに内部トランザクションを外部の顧客にとって利用可能なものにすることを困難にしている。

本発明を、その特定の実施例に関連して詳しく説明したけれども、本発明の精神及び技術的範囲を逸脱することなく、形式及び詳細における上記の及びその他の変更を行い得ることは当業者には明らかであろう。例えば、別の実施例は、SGIクライアント・ルーチン416及びアウトサイド・デーモン312をデーモン314に組み込むことが可能である。これは更に大きなパフォーマンスを提供するであろうが、httpdの実施形態のプロプライエタリ及びその改良版を組み込むことを難しくするであろう。

## 8. 修正の内容

- (1) 請求の範囲の記載を別紙1のとおりに修正する。
- (2) 明細書第3頁第11行の「ことなく内部資源を」を、「ことなく、内部資源を」に修正する。
- (3) 明細書第20頁第10行の「469」を、「464」に修正する。
- (4) 明細書第21頁第15行の「認証ストリング」を、「文字ストリング」に修正する。
- (5) 明細書第26頁第1行の「464」を、「419」に修正する。
- (6) 明細書第26頁第3行の「465」を、「464」に修正する。
- (7) 明細書第26頁第4行の「466」を、「465」に修正する。

## 手続補正書

平成13年 3月12日

特許庁長官 殿

## 1. 事件の表示

平成 9 年 特許願 第 517127 号

## 2. 補正をする者

住 所 アメリカ合衆国10504、ニューヨーク州アーメンク(各地なし)  
名 称 インターナショナル・ビジネス・マシーンス・コーポレーション

## 3. 代理人

生 所 東京都新宿区西新宿七丁目21番1号、新橋ロイヤルビル3階  
山本特許事務所  
電 話 (03) 3366-6730  
FAX (03) 5337-6738  
氏 名 弁護士 山本 仁 樹  
(6545)

## 4. 補正対象書類名

- (1) 明細書
- (2) 請求の範囲

## 5. 補正対象項目名

- (1) 明細書
- (2) 請求の範囲

別紙1

## 請求の範囲

1. 内部コンピュータ・システム(320)と外部コンピュータ・システム(310)との間のセキュリティ・ファイアウォール(300)に違反することなく内部資源(330)を使用して前記外部コンピュータ・システムがトランザクション・リクエスト(2)を開始することを可能にするように前記内部コンピュータ・システムに指示するための方法にして、  
前記内部コンピュータ・システムと前記外部コンピュータ・システムとの間の前記内部コンピュータ・システムによって開始される接続を確立し、それによって認証済み接続を確立するステップ(451)と、  
前記外部コンピュータ・システムが受け取ったトランザクション・リクエストを前記外部コンピュータ・システムによって呼び出すステップ(401)と、  
前記トランザクション・リクエストの呼出しに responding、プロセス環境変数を含むオリジナル・プロセス環境を前記外部コンピュータ・システムによって作成し、前記トランザクション・リクエストを実行するために前記トランザクション・リクエスト及び前記プロセス環境変数を含むストリングを作成するステップと、  
前記認証済み接続を通じて前記内部コンピュータ・システムに前記ストリングを前記外部コンピュータ・システムによって送るステップ(435、436、437)と、  
前記トランザクション・リクエストを前記内部コンピュータ・システムによって検証するステップ(464)と、  
前記オリジナル・プロセス環境を前記内部コンピュータ・システムによって再作成するステップ(471)と、  
前記トランザクション・リクエストを前記内部コンピュータ・システムによって実行し(472)、それによって出力を発生するステップ(482)と、  
を含む方法。
2. 前記確立するステップは、  
(a) 前記外部コンピュータ・システム(310)によって、前記外部コンピュータ・システムにおけるパスワード・ファイルのコミュニケーション・ポートを

## 別紙 1

第 1 コミュニケーション・ポートとして識別し、該パスワード・ファイルからパスワードを第 1 パスワードとして読み出すステップと、

(b) 前記外部コンピュータ・システムによって、前記第 1 コミュニケーション・ポートにおいて第 1 ソケットを作成し、前記内部コンピュータ・システム (320) からの接続ツールを前記第 1 ソケットにおいて接続するステップと、

(c) 前記内部コンピュータ・システム (320) によって、前記内部コンピュータ・システムにおけるパスワード・ファイルのコミュニケーション・ポートを第 2 コミュニケーション・ポートとして識別し、該パスワード・ファイルからパスワードを第 2 パスワードとして読み出すステップと、

(d) 前記内部コンピュータ・システムによって、前記第 2 コミュニケーション・ポートにおいて第 2 ソケットを作成し、前記第 2 ソケットを通して前記外部コンピュータ・システムに接続ツールを送り、それによって接続を確立するステップと、

を含む請求の範囲第 1 項に記載の方法。

3. 前記確立するステップは、

(a) 前記内部コンピュータ・システム (320) によって、前記第 2 ソケットを通して前記外部コンピュータ・システム (310) に固有のタイムスタンプを送るステップと、

(b) 前記外部コンピュータ・システムによって、受け取られたタイムスタンプをもって前記第 1 パスワードをマングルするステップと、

(c) 前記外部コンピュータ・システムによって、前記マングルされた第 1 パスワードを第 1 変換アルゴリズムによって暗号化し、それによって第 1 パスワード・ストリングを作成するステップと、

(d) 前記外部コンピュータ・システムによって、前記第 1 パスワード・ストリングを前記内部コンピュータ・システムに送るステップと、

(e) 前記内部コンピュータ・システムによって、前記第 2 パスワードを使用して、ステップ (c) 乃至 (d) の前記外部コンピュータ・システムにおけるステップと同じステップを反復し、それによって第 2 パスワード・ストリングを作成するステップと、

## 別紙 1

前記ストリングを作成するステップは、前記スクリプトによって前記ストリングを作成するステップを含む、

前記ストリングは前記コマンド、引数、及び前記トランザクション・リクエストを実行するためのプロセス環境変数を更に含む請求の範囲第 6 項に記載の方法。

9. 前記ストリングを作成するステップは、

前記コマンド入力データ、前記外部コンピュータ・システムに常駐する第 2 デモン (312) に接続するためのコミュニケーション・ポート、及びトランザクション・リクエスト (2) のタイプ及び前記ストリングを識別する識別子を前記外部コンピュータ・システム (310) に送って、前記外部コンピュータ・システムに常駐するクライアント・ルーチンを前記スクリプトによって呼び出すステップを更に含む請求の範囲第 7 項に記載の方法。

9. 前記ストリングを作成するステップは、

前記スクリプトによる呼び出しを受けることに応答して、前記クライアント・ルーチンを前記第 2 デモンによって認証するステップと、

前記第 2 デモンの動作を前プロセス及び子プロセスに分離し、前記親プロセスが前記第 1 ソケットにおいて前記内部コンピュータ・システムからの呼び出しを処理するようにするステップと、

前記クライアント・ルーチンを認証することによって、前記内部コンピュータ・システムに常駐する第 3 デモン (322) にトランザクション・リクエストのタイプを前記子プロセスによって送るステップと、

を更に含む請求の範囲第 8 項に記載の方法。

10. 前記検証するステップ (454) は、

前記外部コンピュータ・システムにおけるメモリに記憶された有効サービス・テーブルを前記第 3 デモンによって読み出すステップと、

前記子プロセスから受け取ったトランザクション・リクエストのタイプを前記有効サービス・テーブルと比較するステップと、

を含む、前記タイプが前記有効サービス・テーブルにおいて見つかる場合、前記トランザクション・リクエストが検証される請求の範囲第 9 項に記載の方法。

11. 内部コンピュータ・システム (320) と外部コンピュータ・システム (310) との間のセキュリティ・ファイアウォール (300) に違反することなく内部資源 (330) を使用して前記外部コンピュータ・システムがトランザクション・リクエスト (2) を開始することを可能にするように前記内部コンピュータ・システムに指示するための独自にプログラムされたシステムにして、

## 別紙 1

(j) 前記内部コンピュータ・システムによって、前記第 1 パスワード・ストリングと前記第 2 パスワード・ストリングとを比較するステップと、

を含む請求の範囲第 2 項に記載の方法。

4. 前記マングルするステップは、

前記タイムスタンプを 16 進数 0177 と論理的に AND して固有の結果を生じさせるステップと、

前記固有の結果を前記第 1 パスワードの各文字と論理的に排他的 OR し、それによって前記マングルされた第 1 パスワードを生じさせるステップと、

を含む請求の範囲第 3 項に記載の方法。

5. 前記暗号化するステップは、

前記マングルされた第 2 パスワードの各文字をキーでもって暗号化し、それによって前記第 2 パスワード・ストリングを作成するステップ

を含む請求の範囲第 3 項に記載の方法。

6. 前記呼び出すステップは、入力データ、引数、及びトランザクション・プログラムを実行するためのコマンドを含むトランザクション・リクエスト (2) を前記外部コンピュータ・システム (310) へ外部ネットワークによって送るステップを含む、

前記ストリングを作成するステップは、前記外部コンピュータ・システムが前記トランザクション・リクエストを受け取ることに応答して、前記プロセス環境変数を含むオリジナル・プロセス環境を第 1 デモン (314) によって定義するステップを含む、

請求の範囲第 1 項に記載の方法。

7. 前記呼び出すステップは、

前記トランザクション・リクエスト (2) を前記外部コンピュータ・システム (310) によって呼び出すことに応答して、前記コマンドを呼び出すステップと、

前記コマンドを呼び出すことに応答して、スクリプトを実行し、前記入力データ、引数、及びトランザクション・リクエストを送るステップと、

を含む、

## 別紙 1

310) との間のセキュリティ・ファイアウォール (300) に違反することなく内部資源 (330) を使用して前記外部コンピュータ・システムがトランザクション・リクエスト (2) を開始することを可能にするように前記内部コンピュータ・システムに指示するための独自にプログラムされたシステムにして、

前記内部コンピュータ・システムと前記外部コンピュータ・システムとの間の前記内部コンピュータ・システムによって開始された接続を確立し、それによって認証済み接続を確立するための手段 (312, 322) と、

前記外部コンピュータ・システムが受け取ったトランザクション・リクエストを前記外部コンピュータ・システムによって呼び出すための手段 (314) と、

前記トランザクション・リクエストの呼び出しに反応して、プロセス環境変数を含むオリジナル・プロセス環境を前記外部コンピュータ・システムによって作成し、前記トランザクション・リクエストを実行するために前記トランザクション・リクエスト及び前記プロセス環境変数を含むストリングを作成するための手段 (415, 416) と、

前記認証済み接続を通して前記内部コンピュータ・システムに前記ストリングを前記外部コンピュータ・システムによって送るための手段 (312) と、

前記トランザクション・リクエストを前記内部コンピュータ・システムによって検証するための手段 (322) と、

前記オリジナル・プロセス環境を前記内部コンピュータ・システムによって作成するための手段 (324) と、

前記トランザクション・リクエストを前記内部コンピュータ・システムによって実行し、それによって出力を発生するための手段 (426) と、

を含むシステム。